

# **MICROPROCESSOR BASED CONTROLLER FOR BRIDGE CONVERTERS AND PWM INVERTERS**

**A Thesis Submitted  
In Partial Fulfilment of the Requirements  
for the Degree of  
MASTER OF TECHNOLOGY**

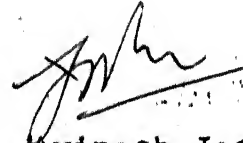
**By  
MUKUND. L. GHONASGI**

**to the  
DEPARTMENT OF ELECTRICAL ENGINEERING  
INDIAN INSTITUTE OF TECHNOLOGY, KANPUR  
FEBRUARY, 1985**

15 JUN 1985  
L.I.T. KANPUR  
CENTRAL LIBRARY  
87541  
Vol. No. A

## CERTIFICATE

This is to certify that the thesis entitled  
'MICROPROCESSOR BASED CONTROLLER FOR BRIDGE CONVERTERS  
AND PWM INVERTERS' is a record of the work carried  
out under my supervision by Shri Mukund L. Ghonasgi  
and to the best of my knowledge it has not been  
submitted elsewhere for a degree.



( Dr. Avinash Joshi )  
Assistant Professor  
Department of Electrical Engineering  
Indian Institute of Technology  
KANPUR

Dedicated to  
My Father, Mother and Brother  
who have been a constant source  
of encouragement.



## ACKNOWLEDGEMENTS

I express my thanks to Dr. A. Joshi for being very considerate and encouraging.

I am grateful to Prof. R.N. Biswas for his help and encouragement especially when things were not so well.

Thanks are due to Research Engineer B.V. Reddy for the lively discussions I had with him. I thank Research Engineer P. Malhotra for providing excellent M.D.S. facilities.

I express my thanks to Mr. R.G. Kale for his help and advices. I thank Mr. PK Mishra for his good company and help during the last few lonely days.

I thank Mr. Anand Barry for his sportive help in taking all the photographs in this thesis.

I thank laboratory incharges particularly S.S. Bhatnagar, Kohle, OP Arora, DN Joshi, Tiwari, RP Singh, CS Singh, G. Singh, Tyagi, Munna and SG Ghorpade.

Last but not the least I express my thanks to my friends VV Deshpande, Lt. PK Dutt, AR Shalu, Durga Prasad, Chakravorty, Murthy, Biswajit, Nandi, Gautam, Satyasankar, Saibal, Senthil, Ratha, Gaya Prasad, MV Govind and many others who made my stay at IIT/K enjoyable.

Finally, I thank Shri J.S. Rawat for his neat typing.

Mukund L. Ghonasgi

### ABSTRACT

Microprocessor based controllers for three phase bridge converters and PWM inverters are presented.

The controller for bridge converter gives equidistant triggering, fast transient response with reduced hardware. This is demonstrated by interfacing with a three phase bridge converter.

A multimode three phase pulsewidth modulator is presented. In this scheme independent control of voltage and frequency is possible. A V/F lookup table has been stored for open loop control. In the low frequency range sinusoidal pulsewidth modulation is selected and in the higher frequency range optimal pulsewidth modulation is implemented. The optimal pulsewidth modulation transits through pulse dropping stage to a final square wave mode at rated frequency in a transient free manner. The output voltage can be controlled in steps of 0.4% and frequency in steps of 0.2 Hz.

The experimental trigger pulse waveforms are presented.

## TABLE OF CONTENTS

	Page
CHAPTER 1 INTRODUCTION	1.1
1.1 Literature Survey of Microprocessor based Controllers for Three Phase Bridge Converters	1.4
1.1.1 Asynchronous triggering scheme	1.5
1.1.2 Absolute triggering scheme	1.7
1.2 Literature Survey of Microprocessor based Controllers for Three Phase Pulsewidth Modulators	1.11
CHAPTER 2 CONTROLLER FOR THREE PHASE BRIDGE CONVERTER	2.1
2.1 Gating requirements for Three Phase Bridge Converter	2.2
2.2 Control Logic for Three Phase Bridge Converter	2.3
2.3 Description of Scheme and Hardware Adopted	2.7
2.4 Software Description	2.12
2.4.1 Lookup tables	2.12
2.4.2 Main program	2.14
2.4.3 Interrupt service subroutine for synchronising interrupt	2.16
2.5 Discussions	2.19
CHAPTER 3 STRATEGIES ADOPTED IN PULSEWIDTH MODULATION	3.1
3.1 Overview of Inverters	3.1

	Page
3.2 Sinusoidal Pulsewidth Modulation (SPWM) Strategy	3.3
3.2.1 Sampling techniques	3.4
3.2.2 Carrier wave generation techniques	3.6
3.2.3 Type of sampling	3.8
3.2.4 Choice of the Ratio ( $f_c/f_m$ )	3.9
3.2.5 Limitations of SPWM at high frequencies	3.11
3.3 Other PWM Strategies	3.12
3.3.1 Optimal PWM	3.12
3.3.2 Harmonic elimination scheme	3.13
3.4 Comparison of PWM Strategies	3.14
CHAPTER 4 MULTIMODE THREE PHASE PULSEWIDTH MODULATOR	4.1
4.1 PWM Inverter Gating Signals	4.1
4.2.1 Strategies adopted in pulse-width modulator	4.3
4.2.2 Schematic of hardware and software structure	4.6
4.3 Implementation of Synchronous SPWM	4.10
4.3.1 Implementation of single phase synchronous SPWM	4.11
4.3.2 Implementation of three phase synchronous SPWM	4.17
4.3.3 Software description of synchronous SPWM	4.21
4.3.4 Gear changing technique implementation	4.24

	Page
4.4 Implementation of Asynchronous SPWM	4.26
4.4.1 Implementation of single phase asynchronous SPWM	4.27
4.4.2 Implementation of three phase asynchronous SPWM	4.29
4.4.3 Software description of asynchronous SPWM	4.30
4.5 Implementation of Optimal PWM	4.32
4.5.1 Implementation of single phase optimal PWM	4.34
4.5.2 Implementation of three phase optimal PWM	4.37
4.5.3 Software description of optimal PWM	4.42
4.6 Requirements and Implementation of Mode Changing	4.47
4.7 Lookup tables required for Three Phase Pulsewidth Modulator	4.53
4.7.1 Notch count lookup tables for optimal PWM	4.53
4.7.2 Voltage-frequency lookup table	4.54
4.7.3 Sine wave lookup table	4.55
4.8 Discussion and Experimental Results	4.56
CHAPTER 5 DESCRIPTION OF EXPERIMENTAL HARDWARE	5.1
5.1 Description of Host Microprocessor Workstation	5.1
5.2 Hardware Details	5.3

	Page
5.2.1 Memory	5.4
5.2.2 Timers	5.5
5.2.3 Ports and latches	5.10
5.2.4 Analog to digital converter with multiplexer	5.15
5.2.5 Variable frequency clock generation	5.16
5.2.6 Interrupt multiplexing	5.19
5.3 Description of Auxillary Hardware developed	5.20
5.3.1 Pulse isolation and amplifica- tion stage	5.20
5.3.2 Optical isolation zero crossing detector	5.21
5.3.3 Analog signal optical isolation	5.22
CHAPTER 6 CONCLUSION	6.1
APPENDIX : PRINTED CIRCUIT DIAGRAM	

## CHAPTER 1

### INTRODUCTION

With the advent of microprocessors and very large scale integration circuits, a whole new era of flexible digital control logic systems are being developed. The designs are becoming modular, simpler. Complex logic operations are being accomplished by software manipulations.

In the particular case of control logic for drive system applications, higher performance levels are being attained with rugged digital control. The entire drive system supervisory functions can possibly be accomplished by faster, higher performance 16-bit microprocessors or a multiprocessor arrangement of 8-bit microprocessors. Few of the supervisory functions that can be incorporated in the control logic are enumerated below.

By monitoring parameters defining the current status of the system and implementing suitable system-management functions, it is possible for a microprocessor based controller to achieve various drive function requirements e.g. controlled startup and shutdown procedure and controlled acceleration and deceleration. Management functions can also accommodate suitable protocols, so that one drive can be interfaced to

others or a central control system. The management functions incorporated depends on the size of the overall system, the nature of application and the performance required.

Monitoring and diagnostic functions can be accomplished. One of the main advantages of a microprocessor based controller is its capability of sampling variables in the system, storing them in memory or performing logical operation(s) upon them. Hence it is possible to identify conditions appropriate for alarm or controlled shutdown. Diagnostic capabilities can be provided to identify the faulted component/components thus reducing repair time.

Finally the heart of the system is the switching/control strategy requirements of the controller. A very high performance controller with modular design and reduced hardware is possible.

In this thesis the switching/control strategy for two power electronic circuits is developed. The controller for three phase bridge converter is presented in Chapter 2. The scheme adopted uses a single phase sensing. The hardware required is reduced. The resulting output pulses are equidistant and the fastest transient response of  $1/6$ th cycle is achievable for any change in  $\alpha$ . The scheme can be used for variable frequency supply.



The principle switching strategies for pulsewidth modulator are presented in Chapter 3. The relative merits and demerits are also elaborated.

The design and implementation of the multimode three phase pulsewidth modulator implemented in this thesis is described in Chapter 4. In the low output frequency range of 0 to 20 Hz, a on-line (real time) carrier by carrier, computation based, regularly sampled sinusoidal pulsewidth modulation is implemented. Above this range, optimal PWM is implemented. The optimal PWM smoothly transits through pulse dropping stage to square wave output waveform at 50 Hz. The square wave mode is implemented till maximum output frequency of 100 Hz.

In the sinusoidal pulsewidth modulation range of 0 to 20 Hz, asynchronous SPWM is implemented for 0 to 5 Hz, synchronous SPWM with ratio 36 for 5 to 10 Hz and synchronous SPWM with ratio 18 for 10 to 20 Hz.

A Hysteresis band of 1 Hz is provided between adjacent modes of operation. Smooth transition between adjacent modes is ensured.

In Chapter 5 the details of the hardware developed to implement the above controllers is described. Provisions have been kept to expand the capabilities of the controllers

implemented. With software changes/development, controllers for a few other power electronic circuits are implementable.

The details of printed circuit boards developed etc. are given in the Appendix.

A brief literature review for both the controllers implemented in this thesis is presented in this chapter.

### 1.1 LITERATURE SURVEY OF MICROPROCESSOR BASED CONTROLLERS FOR THREE PHASE BRIDGE CONVERTERS:

All techniques of implementing a microprocessor based controller for three phase bridge converter exploit certain symmetries existing in the three phase ac supply waveform and the triggering requirements of the three phase bridge converters. Initially these salient symmetries are enumerated. Then the combinations of these symmetries used by few authors are discussed.

1. Synchronisation with the three phase ac supply can be achieved by sensing all the line to line voltage zero crossings or by sensing only one line to line voltage zero crossing and utilising it alone or by sensing only one line to line voltage zero crossing and generating from it the instances of the line to line voltage zero crossing for other combinations.

2. The absolute delay (firing) angle for a thyristor can be measured as the delay angle from a proper synchronising signal to the start of the gating or trigger pulse to the thyristor. In steady state the delay angles for all thyristors are equal.

3. In steady state, the angle between successive gating pulses also known as distance between successive firing (DBSF) is 60 degree, independent of delay angle.

4. The gating pulse sequence is fixed and independent of the delay angle.

5. Given the instantaneous values (status) of the three phase ac supply and the range of the firing angle, the thyristors to be triggered before the next synchronising instant (within the sixty degree interval) are uniquely determined.

To exploit the symmetries given above there are two distinct schemes for implementing a microprocessor based control logic for three phase bridge converters namely asynchronous triggering and absolute delay angle triggering.

#### 1.1.1 Asynchronous Triggering Scheme:

Ref.[1,2,3] are based on this principle, the differences being in hardware used and certain other minor variations.

In asynchronous triggering, property three listed above namely the constant sixty degree distance between successive firing (DBSF) in the steady state is used. The initial delay angle is correctly set by referencing a proper synchronising signal then the successive gating pulses are set after every sixty degree interval in the fixed sequence (property four). The distance between successive firing interval is counted by a independently clocked hardware counter. Only one synchronising signal is required, which is obtained by using a zero crossing detector and determining zero crossing of a line to line voltage waveform. This signal is either polled or connected to interrupt to determine the synchronising instant.

For a change in firing angle  $\alpha$ , the DBSF for one or more triggering is appropriately modulated. If  $\alpha$  increases then the DBSF also increases to accomodate change in  $\alpha$ . For example if  $\alpha = 10$  degree changes to  $\alpha = 20$  degrees, the DBSF changes from 60 degrees to 70 degrees for only one interval. Thus after the controller is properly initialised there is no inherent need for sampling synchronising instants. But due to the unsynchronisation of clock used to clock the DBSF counter with the ac supply voltages the delay angle  $\alpha$  can drift with time. To avoid this usually once in every input cycle (six triggerings) the actual delay angle  $\alpha$  of the firing pulses is determined.

The delay angle is measured using zero crossing of the line to line voltage used for synchronisation, a hardware counter clocked by DBSF counter clock and the firing pulse for a suitable thyristor with suitable hardware gating logic. For example with ZCD of  $V_{ab}$  triggering pulse for  $T_1$  is selected. The counter measures actual firing or delay angle  $\alpha$  with respect to supply voltages. Any error in  $\alpha$  due to asynchronous operation is compensated in the same way as the change in firing angle  $\alpha$ . Thus only two counters along with single phase sensing for synchronisation is required. Thus the hardware becomes fairly simple. But the transient response capability is limited. This is due to the fact that for reduction of  $\alpha$  by greater than sixty degrees, the DBSF has to become negative which is not possible. Hence maximum changes allowed in DBSF is limited.

#### 1.1.2 Absolute Triggering Scheme:

In absolute firing scheme, the synchronisation signal (instant) for each thyristor is sampled before actual firing angle is loaded into delay counter. Thus six synchronisation instants are sampled in every cycle. These synchronisation pulses are connected to interrupt line of the microprocessor.

In Ref.[4] synchronisation is achieved with single phase sensing. This results in equidistant triggering.

The line to line voltage zero crossing of some combination is determined using a zero crossing detector. This is fed to a phase lock loop (PLL) with a divide by six ring counter in the feedback. Thus the PLL output gives six output pulses in every ac cycle. The positive transitions of these pulses correspond to the required synchronising instants for the six thyristors. The ring counter is properly set to reflect the status of the ac waveform by sensing alternate stages of the ring counter. At every synchronising interrupt (occurs every sixty degree interval) the actual delay angle  $\alpha$  (range 0 to 180 degrees) is sent to delay counter, hence three counters are required. Since thyristors connected to each phase cannot conduct together one counter can correspond to thyristors connected to each phase. Upon delay interrupt by polling supply status the gating pulse for proper thyristor is set. The transient response for change in  $\alpha$  is poor as actual delay itself is loaded into the delay counter. So change of  $\alpha$  from 0 degrees to 180 degrees cannot be achieved till two sixty degree intervals (two synchronising interrupts) pass through. The hardware required is also elaborate.

In Ref.[5] synchronisation is achieved by sensing the line to line voltage zero crossings for three line to line voltages, all displaced from each other by 120 degrees.

Monostables are triggered at both positive and negative edges of each ZCD output pulse. The outputs of all the monostables are ORed and connected to interrupt. These pulses give the synchronisation instances. The difficulty of three phase sensing is that equidistant triggering is not ensured. The output state of all the zero crossing detectors are read by the microprocessor via port. This reflects the status of ac supply voltages.

From the control voltage, the absolute delay (firing) angle is determined from an arc-cosine lookup table. The absolute delay angle is split into three ranges and modified delay angle within sixty degree interval determined.

$$\begin{array}{lll}
 \text{range 1} & \alpha^* = \alpha & \text{for } 0 \leq \alpha < 60 \text{ degrees} \\
 \text{range 2} & \alpha^* = \alpha - 60 & \text{for } 60 \leq \alpha < 120 \text{ degrees} \\
 \text{range 3} & \alpha^* = \alpha - 120 & \text{for } 120 \leq \alpha < 180 \text{ degrees}
 \end{array} \quad (1.1)$$

At every synchronising interrupt the modified delay is sent to a delay counter. Since delay sent to the counter is within sixty degrees only one delay counter can be used for all thyristors. Upon delay interrupt depending on the range of the delay angle and the status of ac supply voltages proper thyristors are triggered by setting output pulses at a port.

Since the modified delay sent to the delay counter is sixty degree or less (within next synchronising interrupt) any step change in the delay angle  $\alpha$  can be accomplished within sixty degrees. Thus the controller responds within  $1/6$  cycle.

In Ref.[6] the principle of using a single delay counter by loading modified count into delay counter is used. The differences between Ref.[5] and [6] being in determining range of delay angle and also in method of achieving synchronisation.

Synchronisation is achieved by single phase sensing and phase lock loop (PLL). In the feed back path of PLL a divide by six ring counter in cascade with 8-bit counters are used. The output of the PLL gives a string of pulses at frequency of  $2^8 * 6 * f_s$  ( $f_s$  supply frequency). This is used as a clock to delay counter. Hence even the small error due to unsynchronised clock is absent. The maximum delay of 60 degree is represented by 256 counts (0FF Hex). The input of the six stage ring counter has  $6*f_s$  frequency and each positive transition of the pulse corresponds to a synchronisation instant in supply waveform. This is connected to a positive edge sensitive interrupt of the microprocessor. The output of alternate stages of the ring counter gives the status of supply voltage. Depending on control voltage itself the range of firing angle  $\alpha$  is determined. The modified  $\alpha^*$  from



eqn. (1.1) is retrieved from lookup table. Thus complete 8-bit accuracy is available for  $\alpha^*$ . At synchronisation interrupt depending on control voltage, range is determined and modified  $\alpha^*$  retrieved from arc-cosine table. The modified angle is sent to delay counter. At delay interrupt (will occur before next synchronising interrupt) from the range and the ac supply status the proper triggering pulses are set at an output port.

The hardware required for Ref.[6] is elaborated but performance achieved is good. It gives equidistant triggering with best transient response of 1/6th cycle. It can also accomodate variations in supply frequency.

## 1.2 LITERATURE SURVEY OF MICROPROCESSOR BASED CONTROLLERS FOR THREE PHASE PULSEWIDTH MODULATORS:

Initial attempts at implementing pulsewidth modulator control logic were based on lookup table techniques. In Ref.[9,14] the computational schemes are presented for implementing the sinusoidal pulsewidth modulation similar to that implemented in this thesis. Hence the literature survey is restricted to references [9,14].

In Ref.[9] a few schemes for implementing a micro-processor based controllers for sinusoidal pulsewidth modulation are discussed. These have limitation of output

frequency range. Then the computation scheme of determining pulsewidth for regularly sampled sinusoidal pulsewidth modulation is presented.

The output pulsewidth of one output trigger pulse is given by

$$W_{tw} = \frac{T}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi + 1 \right) \text{ sec.} \quad (1.2)$$

$T$  : carrier period in seconds

$V_S$  : desired output voltage

$V_{SM}$  : maximum output voltage

$\phi$  : sine samples

The carrier interval  $T$  is generated by dividing a fixed frequency clock by variable count. The count is based on the output frequency command. This scheme of constant clocking frequency can only be applied to sinusoidal PWM and the resolution in output frequency is also low. For higher resolution a higher constant clock frequency will be required. This implies higher counts for generating interval  $T$  which makes computations of  $W_{tw}$  more time consuming. The scheme has been implemented for carrier to modulating frequency ratio of 12 and 9. The maximum output frequency is 100 Hz.

In Ref.[14] a multimode high performance pulsewidth modulator is presented. In low frequency a computation based

sinusoidal pulsewidth modulation is implemented and in the higher frequency range 80 Hz to 250 Hz harmonic elimination scheme is implemented.

The technique of computation based sinusoidal pulsewidth modulation makes the pulsewidth count a function<sup>of</sup> only output voltage. The clock to pulsewidth counters is variable, the actual clocking frequency being a function of desired output frequency. The interval T (carrier interval) is generated by dividing a variable frequency clock by N (constant = 256). The output pulsewidth of one pulse in a carrier interval is given by

$$W_{tw} = \frac{N}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi + 1 \right) \text{ counts} \quad (1.3)$$

The variable frequency clock is obtained by programmably dividing a higher frequency clock  $f_1$ . The programmable count  $W_{tc}$  is given by

$$W_{tc} = \frac{f_1}{N \cdot f_o \cdot n} \quad (1.4)$$

where  $f_o$  : output frequency  
 $n$  : carrier/modulating frequency ratio  
 $N$  : constant

Thus the voltage and frequency commands are decoupled. The ratio  $n$  selected is 192 or 96.

Harmonic elimination scheme also requires the decoupling of voltage and frequency effects. Hence a multimode strategy can be developed.

The hardware circuit employed is elaborate. The performance index and resolution in output voltage and frequency achieved is extremely good. Output voltage can be controlled in steps 0.39% and frequency by 0.0017 Hz. The output frequency range is 0 to 250 Hz.

## CHAPTER 2

### CONTROLLER FOR THREE PHASE BRIDGE CONVERTER

Three phase bridge converters are used in high power applications like large d.c. motor drives, HVDC transmission. For the large power handled, very precise and rugged control logic is required. Lately, microprocessor based controllers are becoming popular.

The cost of the power circuit itself being large, a microprocessor based control logic with its enormous flexibility is an asset. The development of fault tolerant system. Incorporation of digital closed loop can be achieved with little hardware changes. A microprocessor based system can give precise, rugged, totally digital control logic with reduced hardware and modular design.

In this chapter, the control logic requirements of a three phase bridge converter are briefly presented. An implementation scheme based majorly on the software capabilities of a microprocessor is presented. The hardware and the software required for the scheme is explained. The features, limitations and further improvements in the scheme are discussed.

### 2.1 GATING REQUIREMENTS FOR THREE PHASE BRIDGE CONVERTER:

The circuit diagram of a three phase bridge converter is shown in Fig. 2.1. The firing angle ( $\alpha$ ) is the delay (in electrical angle) in the point of conduction of a thyristor from the point of conduction of the corresponding uncontrolled device, if all thyristors are replaced by diodes. The three phase line to line voltage waveforms along with  $\alpha=0$  degree points for all thyristors have been shown in Fig. 2.2. Considering that all the devices in Fig. 2.1 to be rectifiers, the combinations of devices conducting together are also listed.

To trigger a thyristor from blocking to conducting state a pulse of short duration at the gate is sufficient. But problem is faced during turn on of a thyristor with short pulse triggering for highly inductive loads or active loads like dc motor. To avoid this long pulse firing is preferred. In long pulse firing, the trigger to a thyristor is applied for the entire desired period of conduction of the thyristor. This pulse is (usually) gated with high frequency carrier before being fed to isolation pulse transformer to avoid saturation of pulse transformer.

The proper sequence of conduction for thyristors in a bridge converter is identical to diode bridge sequence as shown in Fig. 2.2. Thus after every sixty degree interval, a

new thyristor is triggered ON and continues conduction for a 120 degree interval. The sequence of conduction of thyristors is 1,2,3,4,5,6,1... The pairs of thyristors conducting together in an input ac cycle are 1-2, 2-3, 3-4, 4-5, 5-6, 6-1.

The triggering pulses (long pulse) and output voltage waveform assuming continuous conduction for  $\alpha=30$ ,  $\alpha=90$  and  $\alpha=150$  degrees are shown in Fig. 2.3. The average output dc voltage is given by

$$V_o = \frac{3\sqrt{3} V_m \cos\alpha}{\pi} \quad (2.1)$$

For delay angle  $0 \leq \alpha \leq 90$  degrees, the output voltage and output current are positive. Hence output power is positive implying power flow from source to load. For  $90 < \alpha \leq 180$  degrees. The output voltage is negative but current is positive. Hence output power is negative implying power flow from active load to source i.e. regeneration.

## 2.2 CONTROL LOGIC FOR THREE PHASE BRIDGE CONVERTER:

The functional blocks in a control logic for 3 phase bridge converter is presented. A technique of determining the triggering instants for all thyristors using only one counter is presented.

The gating requirements explained, in Sec. 2.1 and further discussion to follow assumes that certain firing

angle ' $\alpha$ ' has to be achieved. But in most systems, it is only the output parameter which is of importance, the output voltage in case of converters. Hence the input command is proportional to output voltage, ' $\alpha$ ' is an intermediate implementing parameter. To achieve linear input to output relationship, it can be seen from eqn. (2.1) that  $\alpha$  is K times inverse cosine of input command, where K is a constant. This is known as inverse cosine firing strategy. Hence the heart of implementation logic is ' $\alpha$ ' control, but controller should be able to accept the output voltage commands.

The firing angle  $\alpha$  of a thyristor is the delay from a fixed  $\alpha=0$  point for the thyristor in the input ac waveform. Hence the control logic should be able to reference these fixed points 'markers' in the input ac waveform to achieve synchronisation. A 3 phase converter logic needs six synchronising signals for the thyristors in a bridge Fig. 2.1. These are shown as  $\phi 1$  to  $\phi 6$  in Fig. 2.4. It can be noticed that the synchronising signals are equally spaced at 60 degree interval and correspond to the zero crossover points of the line to line voltage waveforms. Hence these points can be determined using zero crossing detectors Ref. [5]. This is the three phase sensing scheme. It requires some extra hardware and has the problem of generation of subharmonics (Ref. [1,6]).



In a single phase sensing, only one line to line voltage transitions is sensed using zero crossing detector. From this ZCD pulse the six synchronising signals are generated. This can be achieved by multiplying the frequency of ZCD pulse by 6 using phase locked loop, so that each positive going edge of the output pulse train is a synchronising signal. Ref. [4,6]. Another technique is to determine 60 degree interval from the ZCD pulse duration and then repetitively count down this interval. The terminal count of the 60 degree interval gives the synchronising signals.

By observing the gating pulse requirements in Fig. 2.3, some symmetries can be inferred for steady state operation. In every 60 degree interval independent of ' $\alpha$ ', an incoming or new thyristor begins conduction there by commuting the conducting thyristor in the same group i.e.  $T_1, T_3, T_5$  or  $T_2, T_4, T_6$ . Each thyristor conducts at a stretch for a total period of 120 degrees. Thus the triggering pulse to the outgoing thyristor is withdrawn at the same instant as the triggering pulse is fed to the incoming thyristor. Thus the identification of ' $\alpha$ ' delay instant is required to apply or withdraw the triggering pulse to proper thyristors.

To identify ' $\alpha$ ' instant for a thyristor, a counter should be loaded with delay proportional to ' $\alpha$ ' at a proper

synchronising 'marker' signal. The ' $\alpha$ ' instant is then given by the terminal count. Thus identification of the proper synchronising signal along with a counter is required to generate triggering pulses for a thyristor. But as already stated above, between two synchronising signals a new thyristor starts conduction commuting the conducting thyristor of the same group. This feature is independent of ' $\alpha$ '. Thus if delays loaded into counter are less than sixty degrees for all ' $\alpha$ ', only one counter is sufficient to identify triggering instant between two synchronising signals. To achieve this ' $\alpha$ ' is split into three ranges 0-60, 60-120, 120-180 degrees. The delay loaded into counter for the three ranges is given in eqn. (2.2).

modified delay  $\alpha=(\alpha)$  deg. for  $0 \leq \alpha < 60$  degrees

modified delay  $\alpha=(\alpha-60)$ deg. for  $60 \leq \alpha < 120$  degrees (2.2)

modified delay  $\alpha=(\alpha-120)$ deg. for  $120 \leq \alpha < 180$  degrees

A constant delay of 60-deg. in case of  $60 < \alpha \leq 120$  is automatically added by shifting the reference for  $T_1$  from  $\phi 5$  to  $\phi 4$  (Table 2.1). Thus the triggering instants can be determined. But to implement ' $\alpha$ ' delay, depending on state of input voltages proper thyristors are triggered as explained below.

A knowledge of the status of the input ac voltage (a snapshot of input ac voltages) along with the range of

angle ' $\alpha$ ' is sufficient to uniquely identify the incoming and conducting thyristors within the next 60 degree interval. Thus by polling the status, determining range of ' $\alpha$ ' and properly sending delay to counter at a synchronising signal, the incoming thyristor, outgoing thyristor at next triggering instant is uniquely determined. The table for ac voltage status, range of ' $\alpha$ ' along with incoming thyristors is given in Table 2.1. For the ac voltage status, logic state '1' implies positive phase voltage, logic state '0' implies negative phase voltage. All the possible statuses in a complete cycle are shown in Fig. 2.4, Ref.[5].

Hence in a system which can identify synchronising signals and access the ac supply status, only one delay counter is sufficient to generate the triggering pulses for all thyristors in a bridge converter. There are different techniques of generating synchronising signal and determining ac supply status. The scheme adopted in this thesis is given in the next section.

### 2.3 DESCRIPTION OF SCHEME AND HARDWARE ADOPTED:

The single phase sensing scheme to achieve synchronisation has the advantages of reduced hardware, achieves Equidistant Triggering (Ref. [1,6]) and hence has been selected for implementation in this thesis. Within the single phase

sensing scheme, the strategy of repeatedly counting down 60 degree interval to generate synchronising signals is selected for simplicity of hardware and greater flexibility due to software manipulations. The scheme of single delay counter for determining triggering instants has advantages of reduced hardware and makes the transient response of the controller  $1/6$ th of the input ac period for any variation in  $\alpha$ . That is even a variation of  $\alpha$  from 0 to 180 degrees can be achieved within  $T/6$  seconds, where  $T$  is the period of input ac waveform. This is the best transient response achievable for a change in input command for a 3 phase naturally commutated bridge converter. The microprocessor based control logic implemented in this thesis for three phase bridge converter using the above techniques is described below.

The zero crossing transitions of line voltage say  $V_{ac}$  is obtained from a zero crossing detector. At the positive edge of this pulse, a monostable is triggered. The output pulsewidth is selected to be small (Fig. 2.5). This gives the  $\alpha=0$  point for thyristor  $T_1$  (Fig. 2.2). The duration of the ZCD pulse is counted by another counter. This is the half cycle duration count for input ac voltage. This duration is divided by 3 by the microprocessor to determine the 60 degree interval for ac input. With variations in supply

frequency the 60 degree interval varies. The computation of 60 degree interval is done once in every supply cycle, and used in the earliest possible 60 degree interval.

The 60 degree interval count is fed to another counter 'synchronising counter'. The synchronising counter counts the 60 degree interval, upon reaching terminal count it generates an output pulse before reloading from input buffer. This output pulse stream gives the  $\alpha=0$  or synchronising 'marker' instances for all the thyristors. This is fed as an interrupt signal to the microprocessor.

The clock fed to the synchronising counter is independent of ac input supply. Due to this unsynchronisation there can be a relative sliding between the synchronising counter output signal and input ac voltage. This will result in generation of incorrect synchronising signals. To avoid this the synchronising counter is reset to 60 degree count at every positive transition of ZCD output pulse with the help of ZCD monostable. Thus once in a supply cycle, the synchronising counter is synchronised to the input ac supply of the bridge converter. This prevents the sliding of synchronising signals.

In one cycle of ac supply, there are six synchronising interrupts, these can be identified by a modulo-6 software interrupt counter. After proper initialisation, the value

of the interrupt number represents the status of the input ac voltage. Thus the status of ac supply can be determined by software. To start such a system properly, the first synchronising interrupt to the controller should be from ZCD monoshot output. This clearly identifies the ac supply status and can be set properly. So with proper initialisation, the interrupt numbers are decremented by modulo-6 counter at every synchronising interrupt and correctly represents the status of the ac supply, see Fig. 2.4. The further interrupts in the system may all be obtained from the synchronising counter or one in every cycle may still be obtained from the ZCD monostable synchronising signal.

A delay counter is required to identify the triggering instants between two synchronising signals. The counter is fed delay proportional to  $\alpha$  from eqn. (2.2). Upon reaching the terminal count it causes an interrupt to the microprocessor. A schematic sequence is shown in Fig. 2.5.

Thus three counters are required for implementation, namely the synchronising counter, delay counter and the period counter. The three counters of a programmable interval timer, the Intel 8253 are used for this purpose (Fig. 2.6). Counter 0 of 8253 is used for delay counting, counter 1 for ZCD duration counting and counter 2 as the synchronising

counter. The 1.536 MHz clock to these counters is derived by dividing the microprocessor clock out by 2. The gate of counter 1 is fed by the ZCD pulse and is programmed in rate generator mode. The counter is thus enabled for the ZCD pulse duration. When the counter is read with gate low, the ZCD duration count can be obtained. Counter 0 is programmed for interrupt on terminal count, its gate is always high. Counter 2 is programmed in rate generator mode. The gate of counter 2 is given the  $\bar{Q}$  output of ZCD monostable. Thus for the entire ac cycle gate is high enabling counting. But upon the positive transition of ZCD pulse, the monostable output causes it to synchronise with the ac supply voltage. The monostable output pulse duration is selected to be just sufficient for synchronising the 8253 counter.

There are three sources of interrupts in the system, the ZCD monostable output, the synchronising signals from counter2 and the delay counter interrupt. The delay counter output is connected to RST 5.5 interrupt. For both the ZCD monostable output pulse and synchronising counter output, the duration of output pulse is small. Hence they require edge sensitive interrupts. Hence both are multiplexed to RST 7.5 interrupt with a control line provided to selectively enable them.

The triggering pulses are set at a output port of programmable peripheral interface 8255, port A. The control

signal for multiplexing ZCD monostable output and synchronising signal is obtained from bit 0 of port C.

For accepting input command from user a bipolar successive approximation analog to digital converter, ADC0800 is provided. The digital equivalent can be read via port B of PPI. If desired, the user can also send his input command through a keyboard interfaced to the system.

The block schematic for the controller is shown in Fig. 2.3.

## 2.4 SOFTWARE DESCRIPTION:

The features and structure of the program for controller control logic is described. The lookup tables required for implementing the logic are also discussed.

### 2.4.1 Lookup Tables:

The input command to the controller is the desired output voltage. A linear input-output characteristics is desirable. Hence the controller determines ' $\alpha$ ' or range and delay from the input command with the help of arc cosine lookup table.

There are two techniques for generating the arc cosine table. In one technique, the absolute value of  $\alpha$  is stored in the arc cosine table. The pointer to the table is the



input command. By checking the value of  $\alpha$  retrieved from the table the range and modified delay can be determined as in eqn. (2.2). In the other scheme, the modified delays are stored in arc cosine table. The pointer within the table is the input command. The range of  $\alpha$  is determined from input command. Say variation of  $+V_{\max}$  to  $-V_{\max}$  is represented by  $00H$  to  $FFH$  respectively. Then for  $\alpha=60$  degrees, the value of input command is  $40$  , for  $\alpha=120$  degrees, the value of input command is  $C0H$ . Hence by determining the zone in which the input command lies the range can be determined as in eqn. (2.3).

$00H \leq \text{input command} < 40H$  range 1 represented by  $00H$   
 $40H \leq \text{input command} < C0H$  range 2 represented by  $08H$   
 $C0H \leq \text{input command} < FFH$  range 3 represented by  $10H$

(2.3)

In the first scheme, two bits of  $\alpha$  are required to determine range hence modified delay has accuracy of 6-bits assuming 8-bit values are stored in lookup table. In the second scheme, since modified delay itself is stored in lookup table and range determined from input command, the entire 8-bit accuracy is achievable for the modified delays. Hence the second scheme is selected for storing arc cosine table.

A arc cosine wave has even symmetry. Hence storing delay for control input 1 to  $\emptyset$  i.e. first quadrant is sufficient. The modified delay angle for input command 0 to -1 can be determined by simple software manipulation. Hence the arc cosine table is of 128 locations for 8 bit input command.

Another lookup table stored is the output lookup table. As has been explained in Sec. 2.3, the range and interrupt number uniquely determines the output trigger pulses at the delay interrupt. To avoid execution time delay in software comparison to determine the output to be made, these are stored in a lookup table. At the delay interrupt, the range along with interrupt number is used as a pointer to the table and retrieve the correct output trigger pulses. The contents of this lookup table is given in Table 1.

#### 2.4.2 Main Program:

The scheme implemented for control logic for three phase bridge converter is interrupt based and interrupt driven. The main program deals with initialisation and the keyboard inputs. The flowchart of the main program is given in Fig. 2.7. The software logic required to generate output trigger pulse is all in the interrupt service subroutines for synchronising and delay interrupts.

The main program initialises counters, ports, scratch pad memory locations, initialises a default input if keyboard

selected as input device. It initialises proper control signal to enable ZCD monostable output to cause first RST 7.5 interrupt for proper initialisation. It display a message on CRT, Type K: keyboard, C: ADC, A: Abort and prompts the user to select an input device. Depending on users choice, the program selects proper interrupt service subroutine for RST 7.5. interrupt by placing appropriate address at the interrupt address pointer for RST 7.5 interrupt. Then the interrupts are enabled.

With the selection of ADC as input device, the user is asked to type A: Abort, <ESC>: input device select. If the user types <ESC>, the system continues to work with ADC as input device till a new selection is made. When the user types A, the program disables interrupts, masks the output trigger pulses, prints a abort message and returns to the monitor program of the system.

With the selection of keyboard as input device, the controller continues to function with previous command (either from keyboard or ADC). The main program itself initialises the default command to 80H. It then displays a message Type A: Abort, I: Input, <ESC>: input device select. When user types I, the user is informed the input format and prompted to give input command. It checks format stores in the scratch pad memory before returning to input mode. The flow chart for the main program is given in Fig. 2.7.

### 2.4.3 Interrupt Service Subroutine for Synchronising Interrupt:

The interrupt service subroutines for keyboard as input device or ADC as input device are nearly identical except that in ISS for keyboard the input command is obtained from a scratch pad location set by main program. In ISS for ADC as input device, the input command is obtained from ADC by giving start pulse and waiting till the conversion is complete before reading input command. The other common tasks are explained below. The flow chart of ISS for ADC as input device with minor additions is given in Fig. 2.8. The flow chart for keyboard as input device is given in Fig. 2.9.

In the Interrupt service subroutine for a synchronising interrupt the interrupt number is decremented modulo-6. A check is made to see if the next synchronising interrupt should come from ZCD monostable output. If yes, the control signal is set high, otherwise it is set low to enable the synchronising counter to cause the next synchronising interrupt. The input command is retrieved, the range determined as explained in Sec. 2.4.1, delay determined as explained below.

Now to determine the delay as a number of clock states required, the question arises whether the ac supply frequency to the converter is of fixed frequency or variable frequency. If a fixed frequency supply is assumed, the contents of the

arc cosine delay table itself can have delay in terms of the number of clock states required to generate the desired angle. If a variable frequency supply is assumed, the contents of the arc cosine table is just the desired delay angle. It is converted to number of clocks states depending on supply frequency. This conversion is achieved by multiplying the delay angle with the number of clock periods present in a 60 degree interval of supply frequency, as determined from counting ZCD duration and dividing by 3. This 60 degree interval is as it is required for the synchronising counter, In the program developed, a supply with variations in frequency is assumed. Now the execution time of the interrupt service subroutine introduces dead zone in control. To minimise this, the execution time delay must be minimum. The multiplication routine used for delay computations neglects leading zero's hence has minimum execution time for small delay angles.

Before the delay number is sent to delay counter, the execution time of the ISS should be compensated. This is absolutely essential when the delay time is computed as the execution time for computation is variable and can be slightly large for computation of large delay angles. The execution time delay of interrupt service subroutine is determined from the number of clock states elapsed since the interrupt was caused. This is done by reading the synchronising counter

'on the fly' and subtracting from the 60 degree count. The execution time delay is subtracted from computed delay before being sent to the delay counter. Thus the dead zone of control reduced and accuracy in delay achieved for a variable frequency supply.

The range determined from input command is stored in scratch pad memory. A check is made if the ZCD duration can be read by polling the interrupt number. If no, the interrupts are enabled and program returns from ISS. If yes, the interrupts are enabled so that no interrupt is kept pending, the ZCD pulse duration is read from counter 1. It is divided by 3 to get 60 degree interval which is sent to synchronising counter and stored in scratch pad memory before returning from ISS. The algorithm for routine for division by 3 used is based on the principle of series summation. It requires a small execution time.

It should be noted that even for a fixed frequency supply with very minor or small variations in supply frequency. The 60 degree interval must be determined on-line in real time by the microprocessor and sent to the synchronising counter to ensure equidistant triggering.

#### 2.4.4 Interrupt Service Subroutine for Delay Interrupt:

In the interrupt service subroutine for delay counter interrupt, the range and interrupt number are retrieved from scratch pad to construct a pointer to output lookup table. The triggering signal for the new pair of thyristors, accessed and sent to output port. The delay counter is disabled by loading mode word and then the ISS returns back to interrupted program. The flow chart is given in Fig. 2.9.

#### 2.5 DISCUSSIONS:

The salient features of the controller scheme implemented in this thesis are equidistant triggering, single phase sensing, extremely fast transient response. The scheme is applicable for variable supply frequency. These features have been achieved with reduced hardware and greater software flexibility.

With software changes, modification of arc cosine delay table, output table and interrupt service subroutines, the same hardware can also be used to trigger single phase bridge converter.

The photographs for experimental results is shown from Fig. 2.10 onwards.

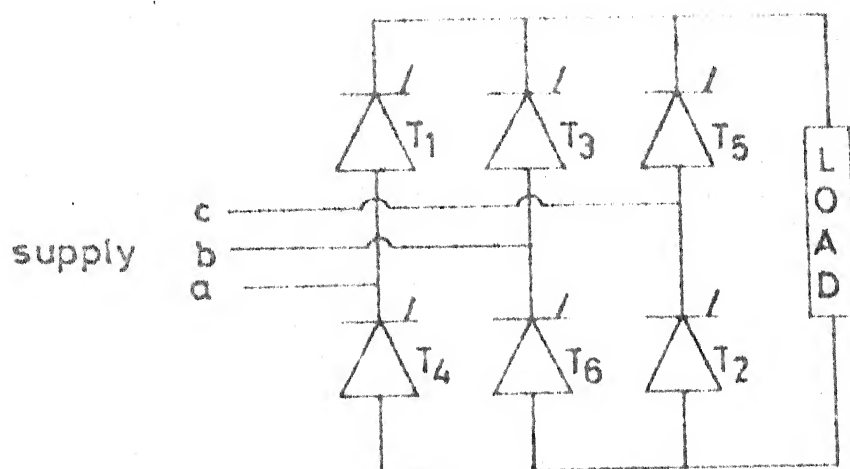
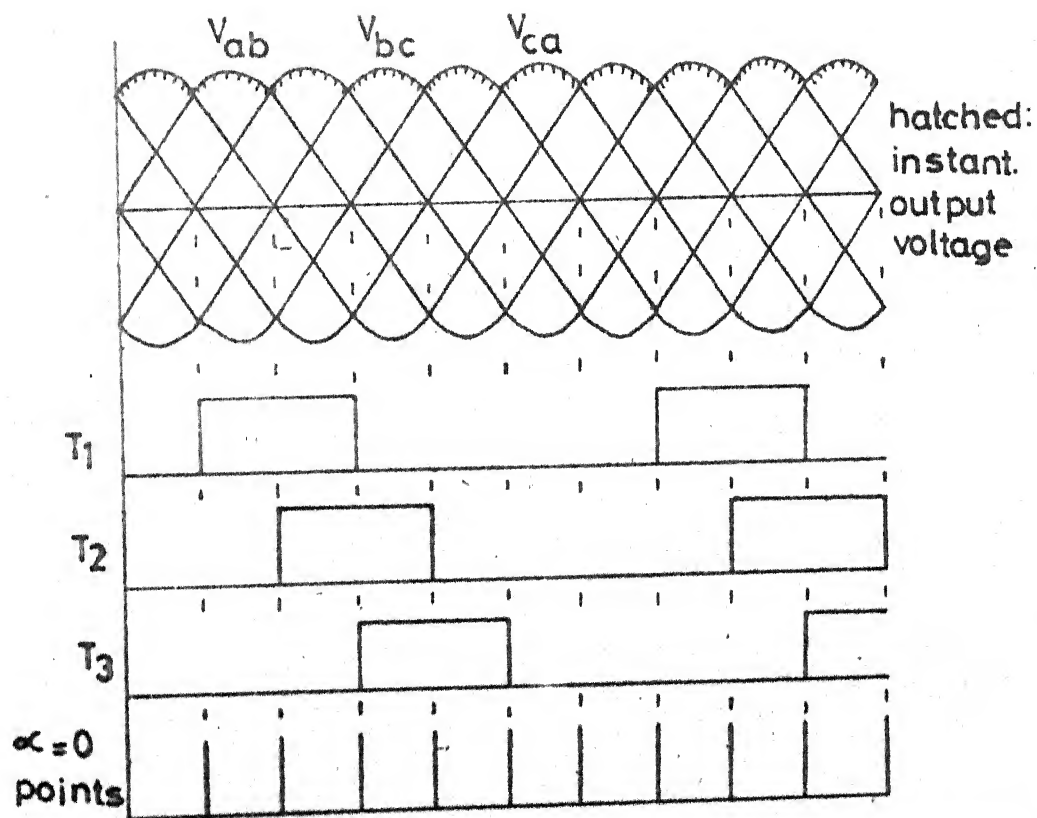


Fig 2.1 Three Phase Bridge Converter

Fig 2.2 Three Phase Voltage with  $\alpha = 0$  points



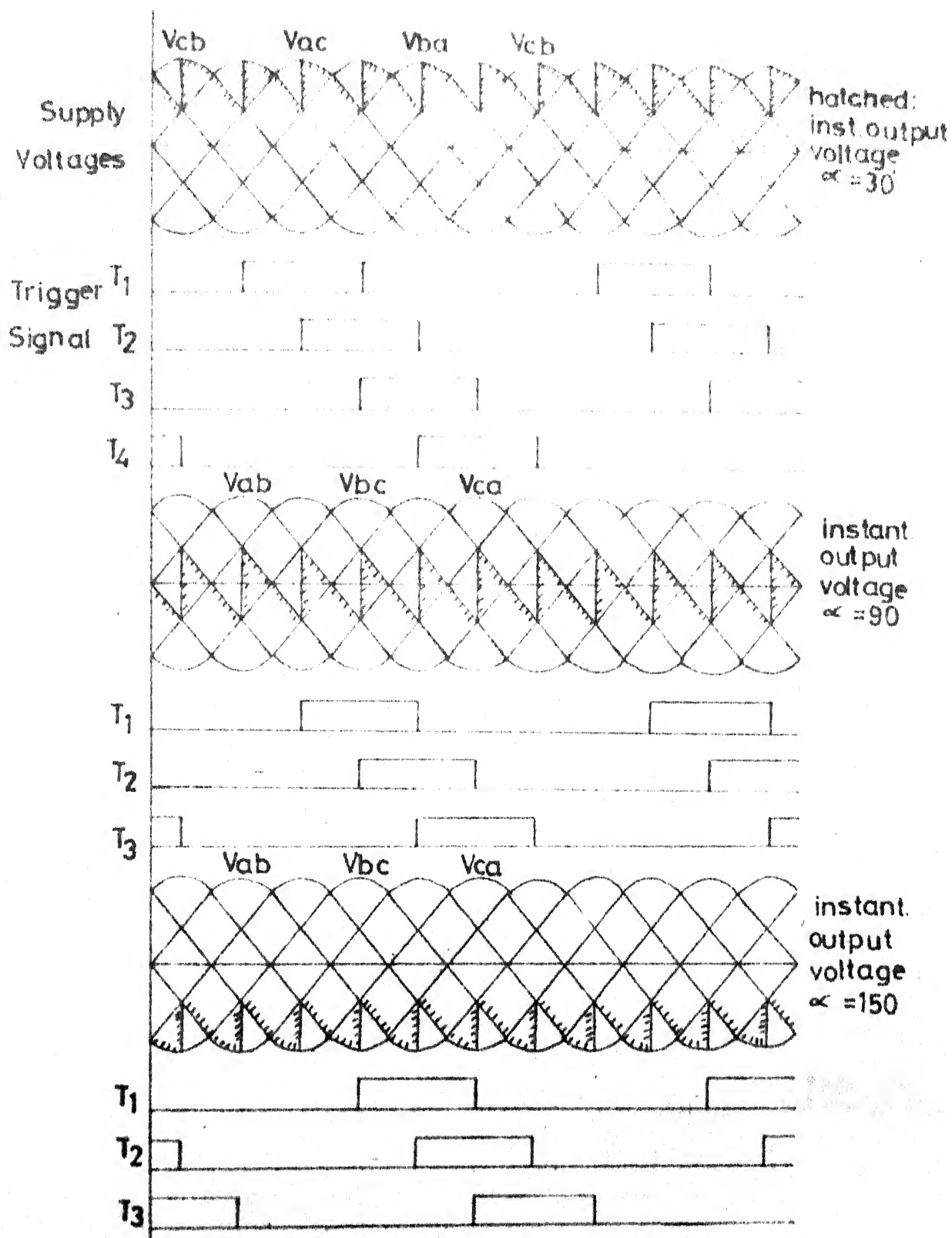


Fig 2.3 Instantaneous Output Voltage, Trigger Pulses  
for  $\alpha = 30, 90, 150$  "assuming continuous conduction"

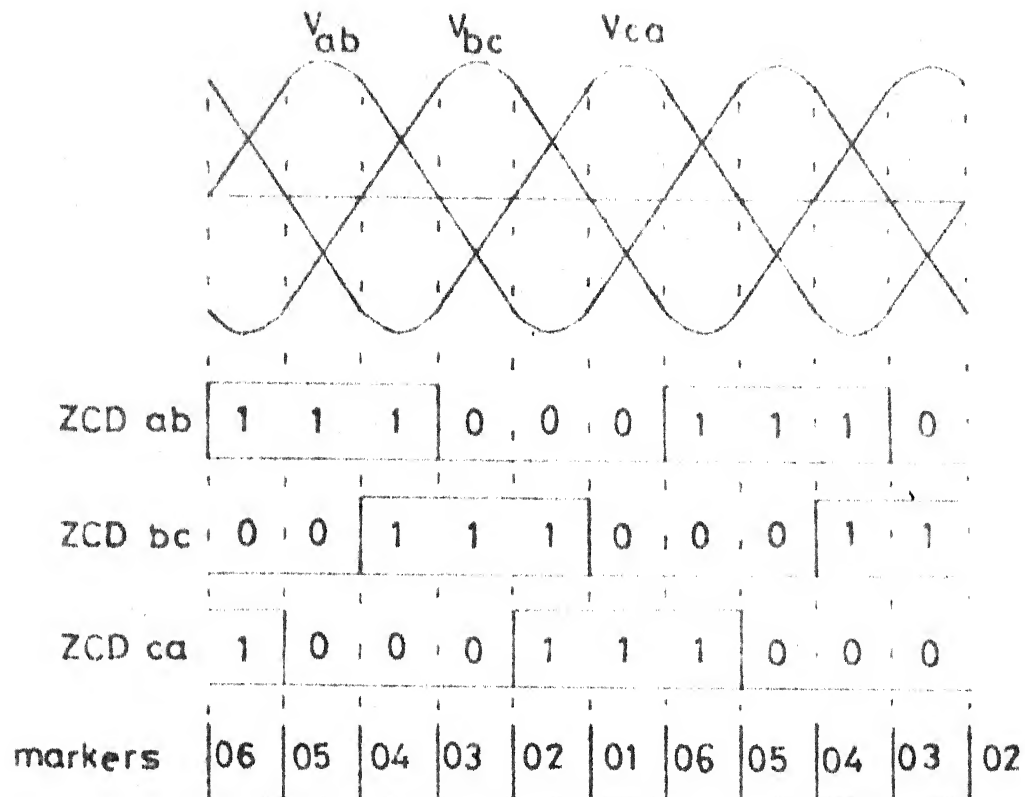


Fig 2.4 Supply Voltages,ZCD pulses,Status,Markers,INTNB

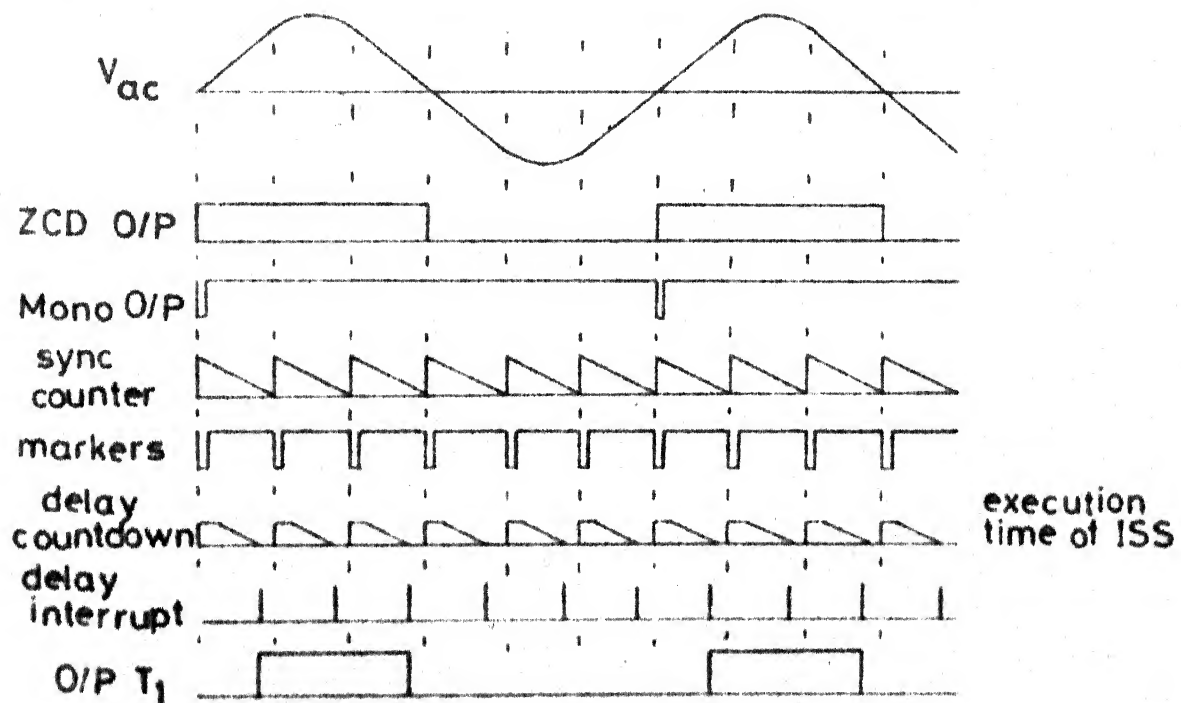


Fig 2.5 A Schematic sequence in the controller

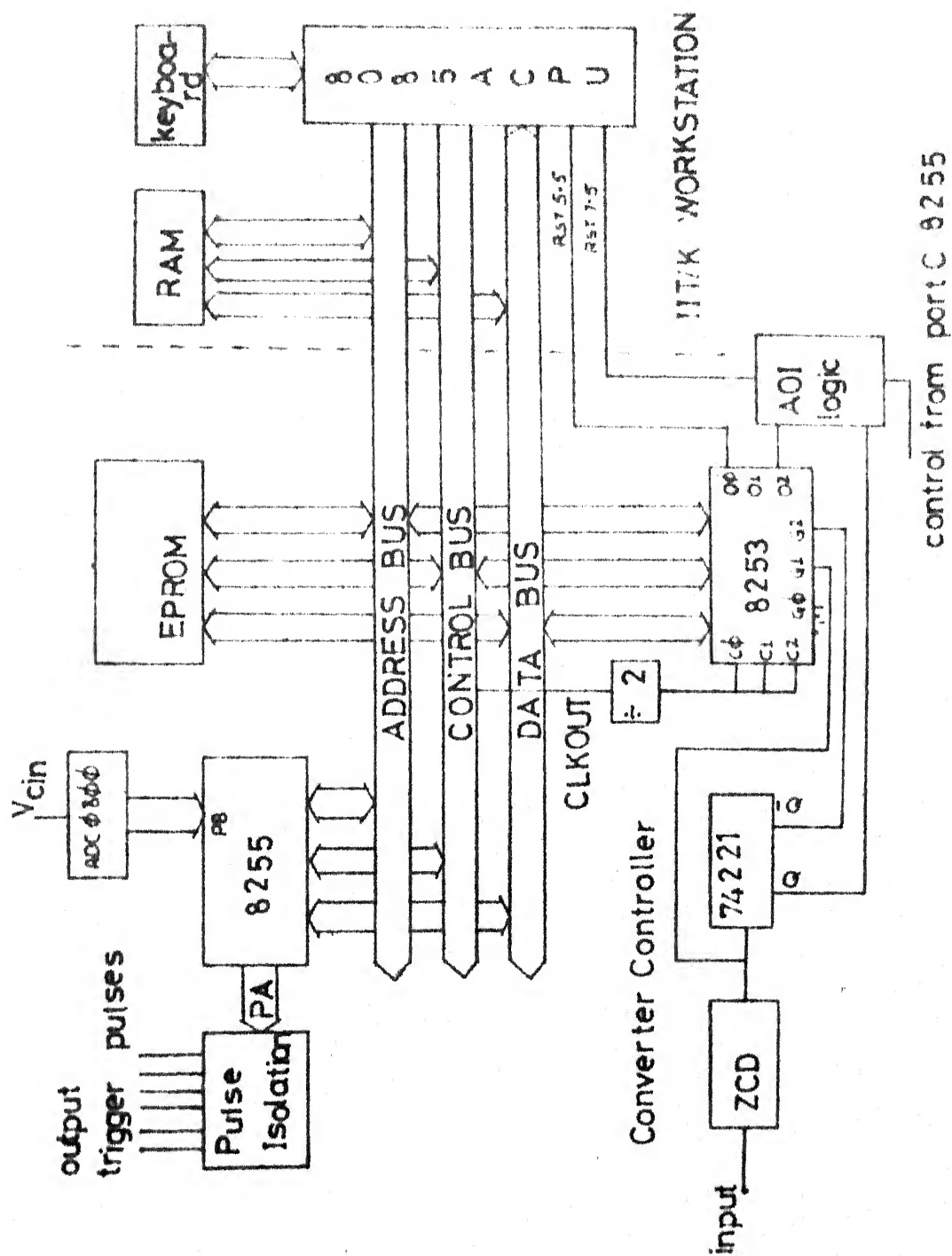


Fig 2.6 Block Schematic of Controller for 3-Phase Bridge Converter

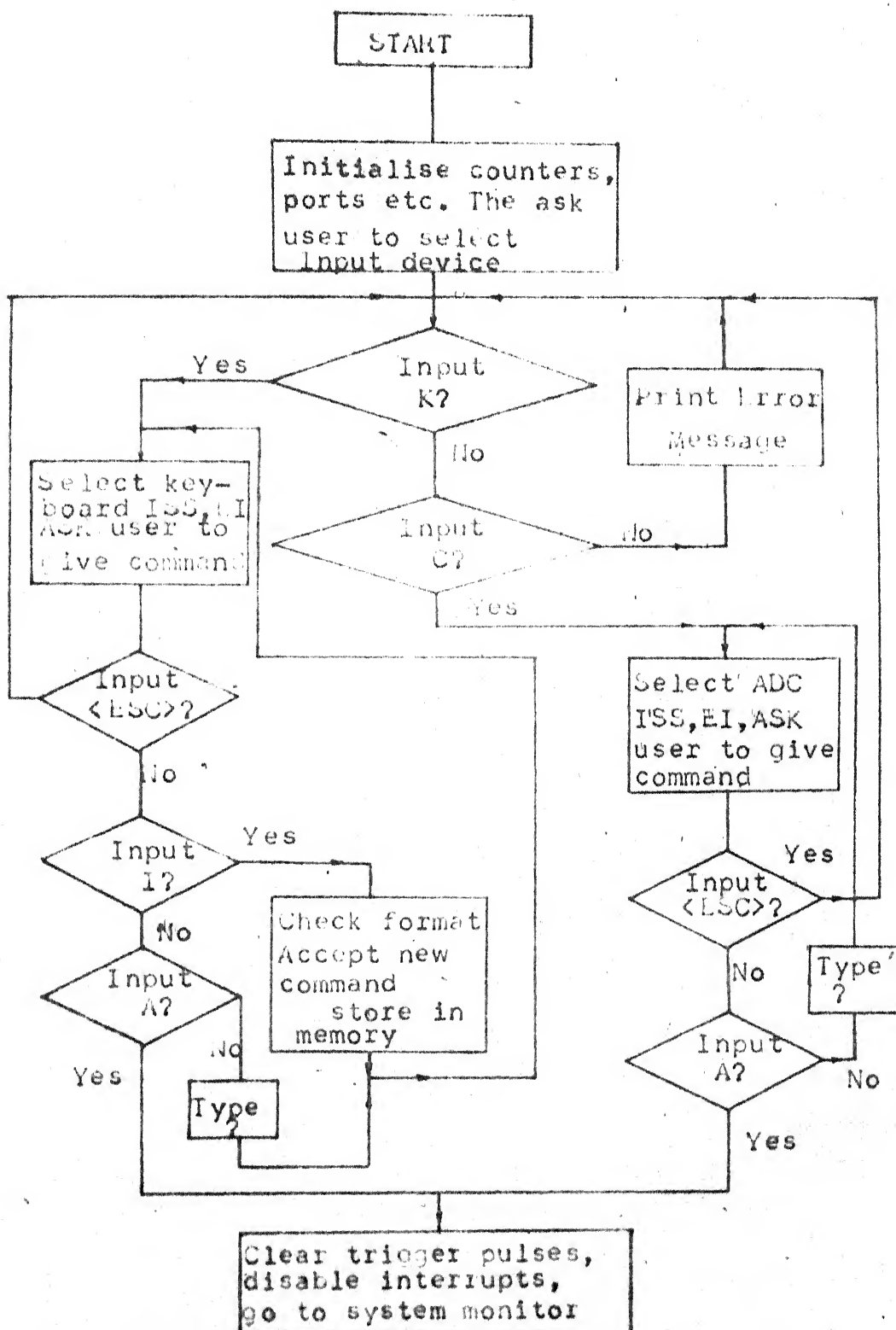


Fig. 2.7: Flowchart for Main Program

Table 1: Thyristors to be Triggered from Range, INTNB

INTNB RANGE	$\phi 6$	$\phi 5$	$\phi 4$	$\phi 3$	$\phi 2$	$\phi 1$
$0 \leq \alpha < 60$ $\phi 0H$	$T_5, T_6$	$T_1, T_6$	$T_1, T_2$	$T_2, T_3$	$T_3, T_4$	$T_4, T_5$
$60 \leq \alpha < 120$ $\phi 8H$	$T_4, T_5$	$T_5, T_6$	$T_1, T_6$	$T_1, T_2$	$T_2, T_3$	$T_3, T_4$
$120 \leq \alpha < 180$ $1\phi H$	$T_3, T_4$	$T_4, T_5$	$T_5, T_6$	$T_1, T_6$	$T_1, T_2$	$T_2, T_3$

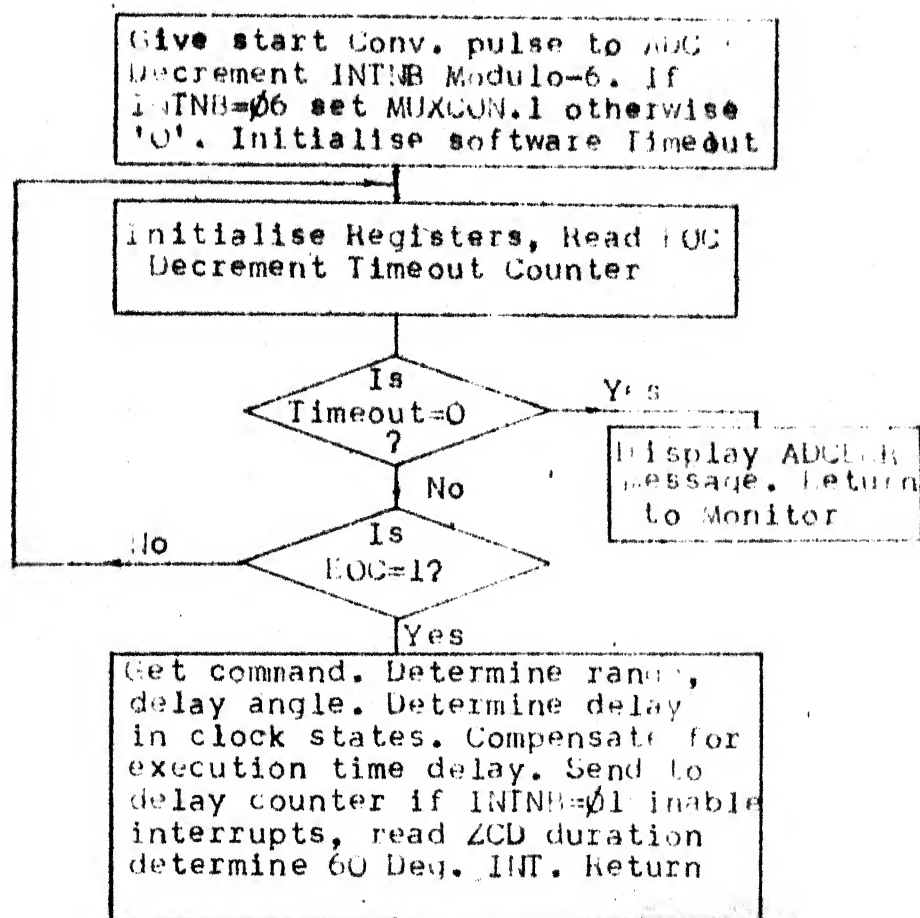


Fig. 2.8: Flow chart of ISS for ADC as Input Device

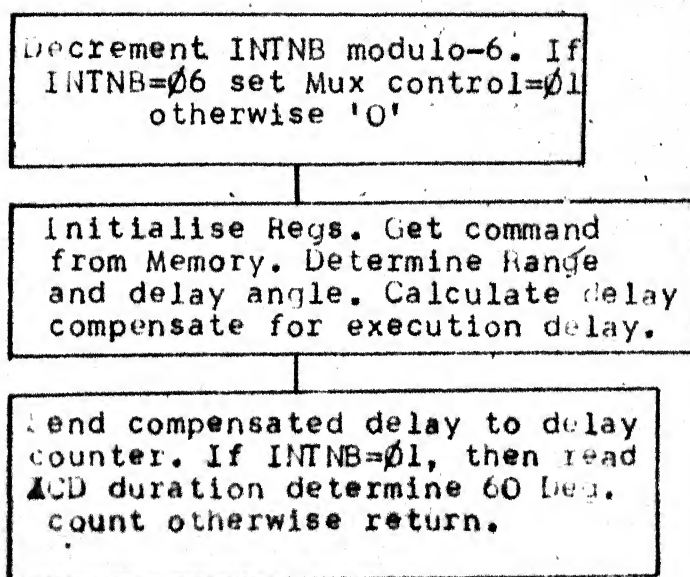
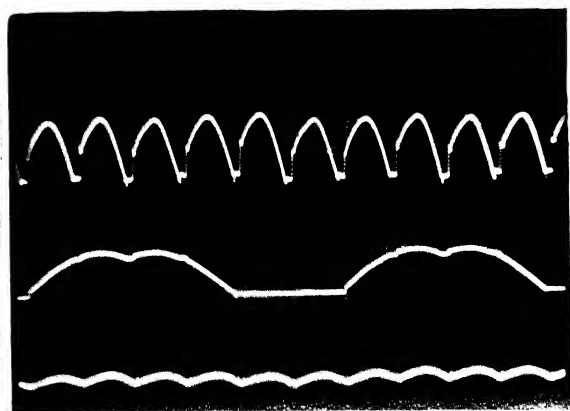
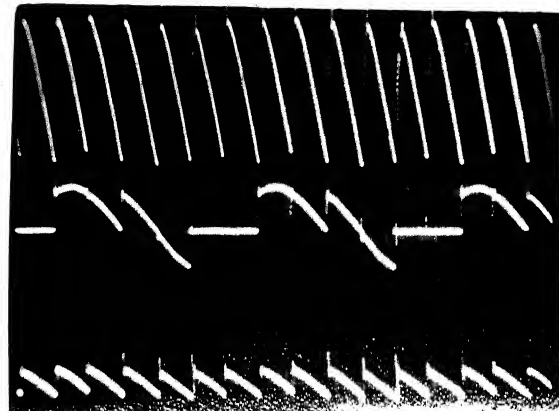
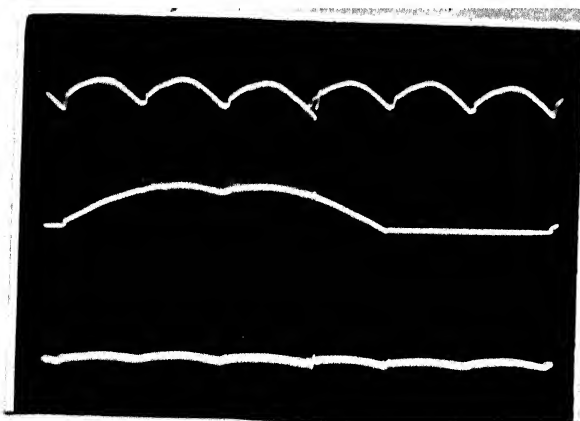


Fig. 2.9: Flow chart ISS for keyboard as Input Device.

a)  $\alpha = 0$ b)  $\alpha = 30^\circ$ 

(time base: 5 msec/cm)

Output voltage, voltage across thyristor and load current R-load

c)  $\alpha = 0$ 

(time base 2 msec/cm)

Output voltage, voltage across thyristor, load current RL load.

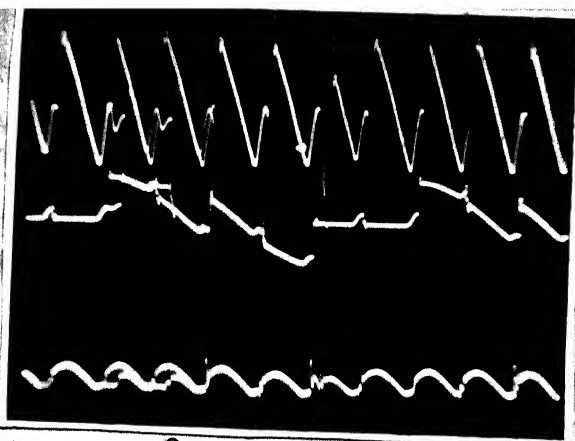
d)  $\alpha = 90^\circ$ 

Fig. 2.10: Photographs of converter output waveforms

## CHAPTER 3

### PULSE WIDTH MODULATION STRATEGIES

In this chapter a very brief overview of different inverters is presented. The various strategies for pulse width modulation along with their merits/demerits are presented.

#### 3.1 BRIEF OVERVIEW OF INVERTERS

With the advancements achieved in low power integrated circuit technology, ushering in of the microchip era, along with the development of high power, high speed semiconductor switches, solid state control of alternating current machinery has gained importance.

For variable speed control of induction motor over wide speed range, inverter fed induction motor drives are preferred. In this both the current fed inverter and the voltage fed inverter are used. To have best control characteristics, independent control of frequency and current, voltage is desirable, depending on type of inverter namely current source inverter, voltage source inverter respectively.

Current source inverter is essentially a two stage power controller. The front end converter is made to operate as current source, along with a current source inverter. This



set up requires a large inductor in the dc link to reduce the ripple in the dc link current. The output of the current source inverter is a quasi-sinusoidal (bipolar square wave) of current. This leads to torque pulsations which are predominant at low speeds. The presence of large inductance in the dc link increases size, leads to slow transient response. Hence current source inverters are not preferred for wide speed, fast response ac drive systems.

The voltage source inverters are of two types, the pulsewidth modulated inverter, inverter with front end converter. The output of a voltage source inverter with front end converter is bipolar square or bipolar stepped wave of voltage. The current is more sinusoidal due to the filtering effect of motor inductance. Hence the torque harmonics are reduced. The control of the output frequency is accomplished by changing the frequency of inverter gating. Change in output voltage is achieved by changing voltage fed to the inverter by properly controlling the converter gating. At low input voltages, commutation problem arise in the inverter. The number of power stages being two the efficiency is reduced. Also each inverter requires an independent converter.

The pulsewidth modulated inverter is a single stage power controller. It requires constant input voltage and

delivers variable voltage, variable frequency output. This is achieved by properly modulating the output pulses of the inverter. Larger number of commutations are required per output cycle increasing commutation losses and reducing reliability. Since these inverters require constant input voltage many units can be operated in parallel with same dc source. The input voltage being constant there is no commutation problem at low output voltages. The input to PWM inverter being constant voltage, uncontrolled sources such as diode bridge, battery can be used. The only deficiencies are large commutation losses, reduced reliability and complex control logic requirements.

With the advent of high power gate turn off thyristors (GTO's) and high power transistors, problem of commutation losses, reduced reliability can be circumvented. With the advent of powerful microprocessors, the complex control logic requirements are accomplished through software manipulations. This makes control logic modular with reduced hardware and high reliability.

### 3.2 SINUSOIDAL PULSEWIDTH MODULATION (SPWM) STRATEGY:

In SPWM strategy, the modulating signal is a sine wave of the desired voltage and frequency. The output pulses are generated by comparison of the modulating signal with the

triangular carrier signal. The output pulses so generated have low harmonic content. The different aspects in SPWM are the generation of the carrier wave, the method of sampling the modulating wave and the type of sampling.

### 3.2.1 Sampling Techniques:

#### 3.2.1.1 Natural SPWM:

In a natural SPWM the switching instances in the output waveform are determined by the points of intersection of the triangular carrier wave with the continuous or unsampled sinusoidal modulating wave as shown in Fig. 3.1. The scheme is best suited for analog control logic. It consists of the triangular carrier wave generator, sinusoidal modulating wave generator and comparator to determine the points of intersection. The design of a sine wave generator for PWM application is elaborated and described in Ref. [13].

The on-line computation of switching instances (in real time) in natural SPWM require solution of transcendental equation. The solution of such equations is beyond the capabilities of present day 8-bit microprocessor Ref.[9].

One way of circumventing this problem is by storing the switching instances in a lookup table. During the operation of the pulsewidth modulator the delays are retrieved and loaded

into the counters sequentially to get the output switching pattern. The size of the lookup table becomes prohibitively large if a fine control over the output voltage is required Ref. [17].

Another way is generate the triangular carrier wave, sinusoidal modulating wave of desired frequencies with a microprocessor and D/A conversion with filters. Comparing them in hardware to generate PWM output Ref. [9].

Due to the limitations mentioned above this has not been implemented in this thesis, although it has minimum harmonic contents. The further discussion of SPWM in this section like carrier wave generation technique, even though valid for the natural SPWM have been explained with reference to regularly sampled SPWM.

#### 3.2.1.2 Regularly sampled SPWM:

In a regularly sampled SPWM the switching instances in the output waveform are determined by the points of intersection of the triangular carrier wave with the sampled modulating wave as shown in Fig. 3.2.

This technique is suited for digital implementation. The switching instances in the output waveform can be determined from eqn. (3.1) [Refer to Fig. 3.2)] Ref. [9]

$$W_{tw} = \frac{T}{2} [ 1 + M \cdot \sin \omega_m t_i ] \quad (3.1)$$

The time instant  $t_i$  is the start of a triangular carrier wave. Eqn. (3.1) can be solved and the switching instances determined with the computational capabilities of a microprocessor. The sine values are obtained from a lookup table. The PWM output is generated by loading into counter the intervals for the positive or negative instantaneous output voltages in one carrier interval.

The samples of a unity sine wave are stored in a lookup table. These samples are scanned with desired step size and frequency to generate the intervals. Hence the logic becomes totally digital.

### 3.2.2 Carrier wave Generation Techniques:

The PWM output is generated by the comparison of modulating signal with the higher frequency triangular wave. With the advent of microprocessor, the actual generation of triangular carrier wave and comparison with modulating signal is not required rather eqn. (3.1) is solved, the pulsewidths in a carrier determined and sent to counter to generate delays. The more important parameter is the carrier frequency or carrier interval. The two techniques for determining carrier frequency are described below.

### 3.2.2.1 Synchronous SPWM:

In synchronous SPWM the ratio of carrier frequency to modulating frequency is held constant. It is equal to the number of pulses in one output cycle, for example 12 in Fig. 3.3. Usually the ratio is selected to be a multiple of three to avoid triplen harmonics in the three phase output. The harmonics in output voltage decreases as the ratio is increased.

The carrier frequency varies with the modulating frequency hence no harmonics lower than the fundamental frequency are generated. In the regularly sampled SPWM the sampling frequency equals carrier frequency.

### 3.2.2.2 Asynchronous SPWM:

In asynchronous SPWM, the carrier frequency is independent of the modulating frequency. Hence non-integer ratio of carrier to modulating frequencies is unavoidable if fine variation of output frequency is desired Fig. 3.4. This leads to problems of beats in the PWM output Ref. [18].

In asynchronous mode the carrier frequency is selected to be the maximum permissible value. The lowest harmonic generated apart from beats is the carrier frequency harmonic. In asynchronous SPWM the sampling frequency is independent of the carrier frequency.

### 3.2.3 Type of Sampling:

#### 3.2.3.1 Symmetric sampling:

In symmetric sampling, the modulating wave is sampled at the start of a carrier and held throughout the carrier interval. The start of the carrier cycle is the peak of the triangular wave Fig. 3.5.

Both the edges of the resulting output pulse are symmetric with respect to the trough of the carrier wave. The edges of the output pulse are said to be modulated equally.

#### 3.2.3.2 Asymmetric sampling:

In asymmetric sampling, the modulating wave is sampled twice in a carrier, once at the start of a carrier and once in the middle of a carrier. The middle of a carrier corresponds to the trough of the triangular carrier wave Fig. 3.6.

Hence the leading and trailing edges of each resulting pulse are determined by two samples of the modulating wave. Therefore each edge is modulated by a different amount. The resulting output pulse is asymmetric with respect to the trough of the carrier wave.

In asymmetric sampling the harmonic contents in the output is smaller compared to symmetric sampling. But then sampling frequency is doubled. For microprocessor implementation where computational technique is used to determine the

pulsewidths, larger number of computations are required for asymmetric sampling. Hence it is usually not preferred.

#### 3.2.4 Choice of the Ratio (Carrier Frequency/Modulating Frequency):

The selection of the ratio  $f_c/f_m$  for synchronous SPWM is discussed.

The higher the ratio  $f_c/f_m$ , the lower the harmonic content in the output waveform. For large values of  $f_c/f_m$ , two major difficulties are encountered. As the modulating frequency increases the commutations/second increase. This leads to the requirement of high speed commutation circuits. High speed commutation circuits require larger rating components and the commutation losses increase. In an induction motor drive system the required voltage increases with the increase in frequency to avoid saturation of the magnetic field. This has a doubly adverse effect. The carrier interval as well as the complementary switch pulsewidth reduces. For proper commutation a minimum pulsewidth is required to trigger the complementary switch. Thus extra hardware/software means have to be provided to ensure the minimum pulsewidth requirement.

At extremely low frequencies the carrier interval is very large. At such frequencies a large  $f_c/f_m$  ratio is required to avoid lower frequency harmonics.



Thus for a synchronous SPWM to operate for a moderate frequency range, it is necessary to change the ratio of carrier to modulating frequency whenever the carrier frequency becomes either too high or too low. This is the ratio changing or gear changing technique Ref. [12].

In analog control logic the implementation of ratio changing increases its complexity. It is with the advent of digital especially microprocessor based control logic that the ratio changing technique has become popular.

Other method for extending the operating range of SPWM is by changing from an asynchronous mode to synchronous mode as the output frequency increases Ref [14,18]. At very low frequencies, if synchronous mode SPWM with moderate ratio is implemented, the harmonics which are multiples of output frequency themselves are of low frequencies. Filtering these harmonics is difficult. In asynchronous mode SPWM, the carrier frequency is held constant at the maximum permissible value. Hence at lower frequencies, larger number of output pulses are generated. Hence changing from asynchronous mode to synchronous mode as output frequency increases, extends the SPWM operating range. In such systems, the change of modes must occur below an output frequency at which subharmonics in the asynchronous PWM waveform cause the motor performance to deteriorate significantly.

### 3.2.5 Limitations of SPWM at High Frequencies:

The serious limitations of implementing SPWM at high frequencies are enumerated.

The maximum output voltage is achieved with square wave output waveshape. To obtain this in SPWM the modulation index has to be infinity which is not possible. This limitation poses problem when the constant dc supply to PWM inverter is derived with uncontrolled rectifier and ac mains supply. The maximum fundamental voltage that can be derived from it with SPWM is lesser than the input mains line voltage. Hence the voltage rating of the motor used in drive system must be lower than the ac supply voltage, if the motor is to be capable of delivering full power at mains frequency. Ref.[10,11]. A way of overcoming this is by pulse dropping, but this leads to voltage transients and elaborate logic is required to implement it.

Depending on the commutation circuit, a minimum pulse-width is required to ensure proper commutation. In SPWM for large modulation index, extra hardware or software logic is required to ensure these requirements.

For better harmonic reduction in output voltage, the ratio of carrier frequency to modulating frequency has to be large. But at higher output frequencies, the number of

commutations/second increase, requiring high speed commutation circuits. These lead to higher losses resulting in reduced efficiency. With the use of high power transistors and high power gate turn off thyristors (GTO's) the problem can be overcome. Another method to extend SPWM range, is the implementation of ratio changing. But very low ratios specially below 9 are not recommended.

### 3.3 OTHER PWM STRATERGIES:

With the advent of digital electronics, PWM techniques suited only for digital implementation have been developed. These are pattern retrieval, lookup table based schemes. The delays in a pattern as well as the pattern itself are stored in lookup tables. During the operation of the pulsewidth modulator, the delays are accessed sequentially and desired output pattern generated.

#### 3.3.1 Optimal Pulsewidth Modulation:

In optimal PWM a predetermined, optimised output pattern with desirable characteristics is generated. A symmetric pattern with certain notches/quarter cycle is selected. The larger the notches/quarter cycle better the output waveform but more commutations are required every cycle.

A suitable performance index is selected and optimised by varying the switching instances in the pattern. The net result of optimisation is the accurate determination of the switching instances. The switching instances have to be determined for each output voltage value.

The performance criteria usually selected are either minimisation of harmonics in output voltage or minimisation of losses in the PWM inverter-induction motor drive system Ref. [15]. The second is usually preferred for induction motor drive system. An optimised pattern with 3 notches/quarter cycle, along with variation in switching angle with output voltage is shown in Fig. 3.7.

#### 3.3.2 Harmonic Elimination in PWM:

In harmonic elimination scheme, undesirable harmonics can be selectively eliminated from the output voltage. Usually dominant lower order harmonics are eliminated by this technique.

The lower order harmonics which are undesirable in the output voltage are decided. Upon this is based the number of notches/quarter cycle. Hence the PWM pattern is decided. Then for each output voltage value the switching instances are accurately determined. These are scaled and stored in delay lookup table ref.[19,20,14].

In harmonic elimination scheme, the desired lower order harmonics are eliminated but no consideration is given to the entire drive system performance.

#### 3.4 COMPARISON OF PWM STRATEGIES:

For an induction motor drive system, the voltage requirements are low for low frequency operation. This is to avoid saturation of the magnetic field. The problem of torque pulsation is predominant at low speeds of the motor, implying low output frequencies of the inverter. Torque pulsations are caused by the interaction of harmonic currents in stator and the magnetic field. To reduce torque pulsations, the harmonics in the stator current must be reduced.

Sinusoidal pulsewidth modulation has minimum harmonics. Also at low voltages the problems of minimum pulsewidth, pulse dropping donot arise. In optimal PWM all harmonics exist, only the performance index selected is optimised. Hence it is not very suited for low frequencies. In harmonic elimination technique, only few lower order harmonics are eliminated. If a large number of harmonics have to be eliminated then the notches/quarter cycle increase drastically along with the delays lookup table size. In SPWM, only sine lookup table is required. Hence SPWM is preferred for low frequency operation.

At higher output frequencies, the motor inductance itself filters out the harmonics in the current. To reduce the commutation losses, the number of switching per cycle should be reduced. The minimum pulsewidth requirements catered and smooth transition to square wave mode should be possible.

The optimal PWM is a pattern retrieval method. When a notch width reduces to lower than required by the power circuit, it is dropped from the pattern and notch widths for the new pattern are calculated. With software manipulations a smooth transition from the optimal PWM through pulse dropping to squarewave mode can be achieved. Usually optimal PWM pattern with two or three notches/quarter cycle are preferred, the commutations/second are reduced. The harmonic eliminated PWM also being a lookup table and pattern retrieval technique, easy transition to square wave is possible with software manipulations. The number of switchings per cycle is slightly large for eliminating moderate number of harmonics. SPWM limitations at high frequencies have already been elaborated in Sec. 3.2.5. Hence the selection is between optimal PWM and harmonic elimination for high output frequencies. In an induction motor drive, since optimal PWM optimises the drive performance as a whole it is preferred ref[15].

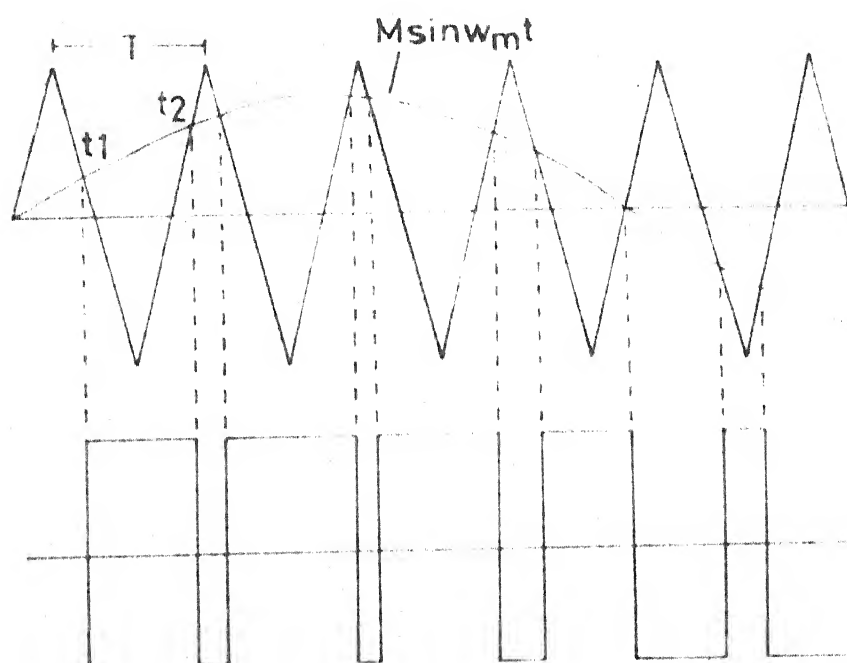


Fig 31 Natural SPWM

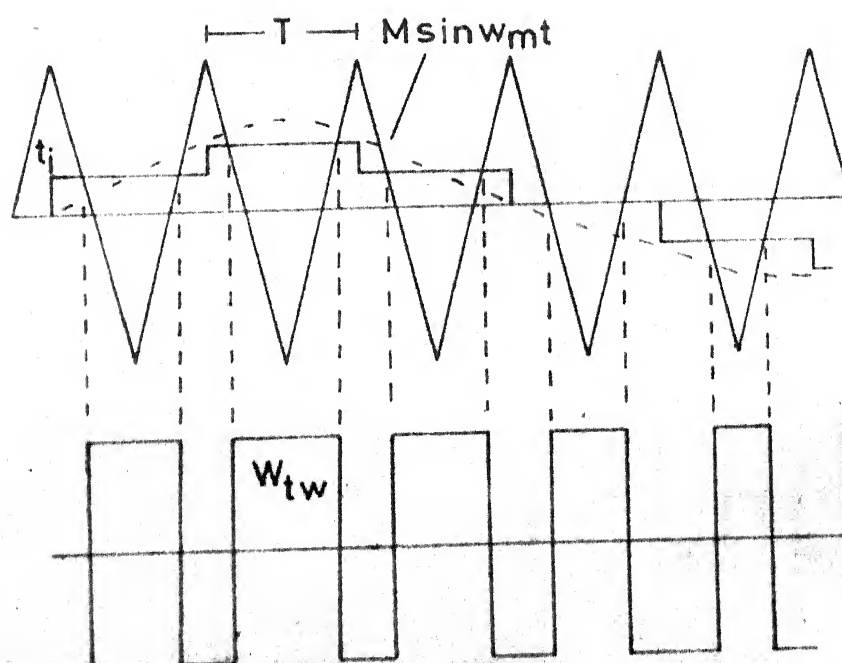


Fig 3.2 Regularly Sampled SPWM

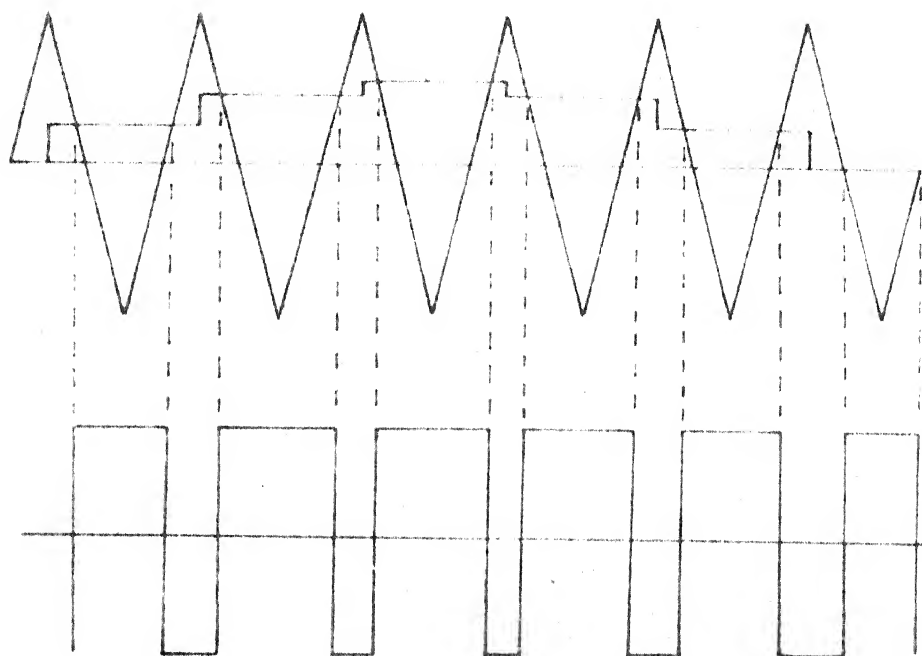


Fig 3.3 Symmetric, Synchronous regular SPWM

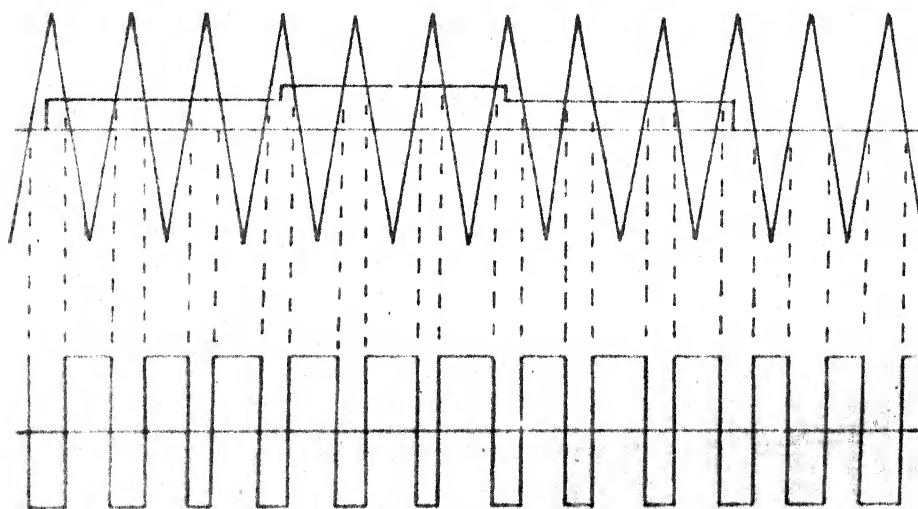


Fig 3.4 Asynchronous regular SPWM



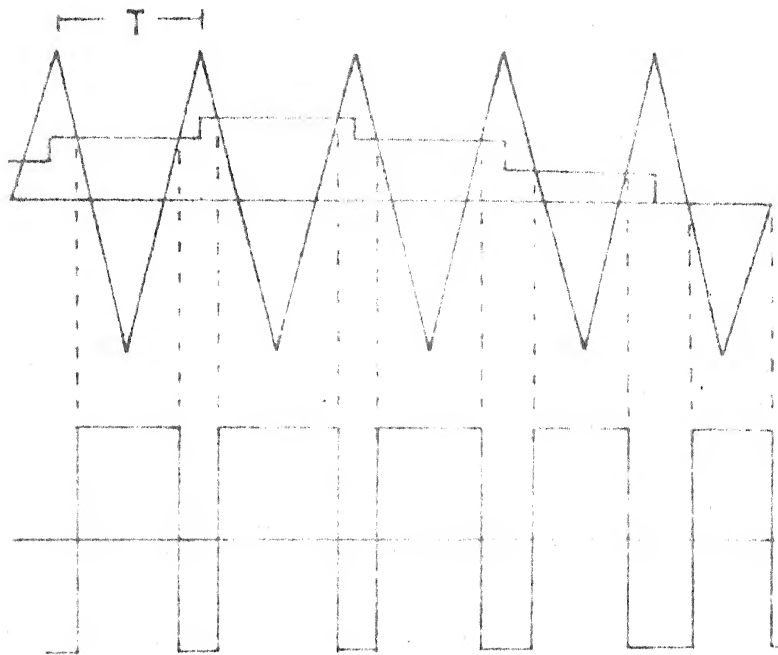


Fig 3.5 Symmetric Sampling

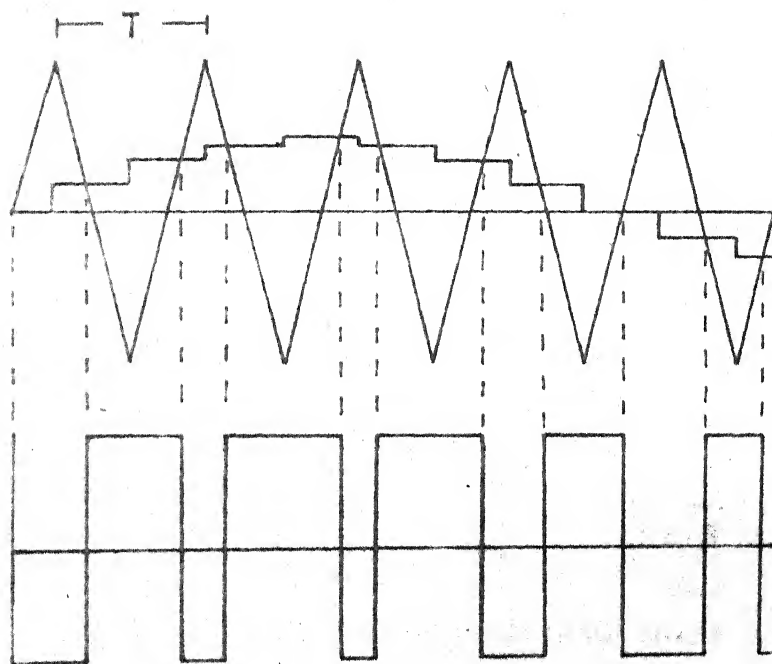


Fig 3.6 Asymmetric Sampling

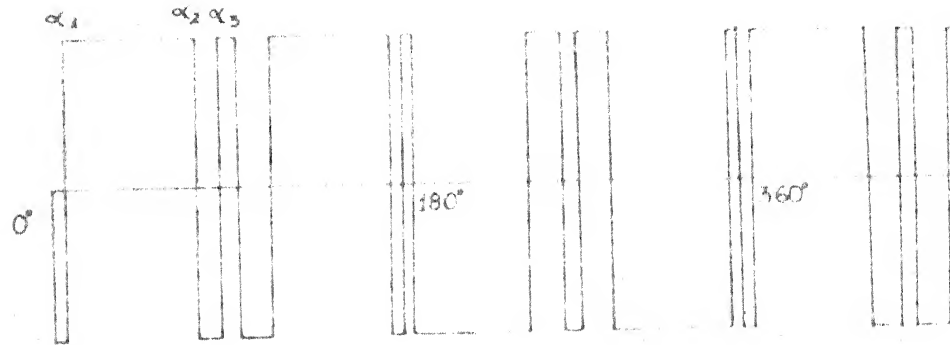


Fig 3.7a Optimal PWM 3 switchings/quarter cycle

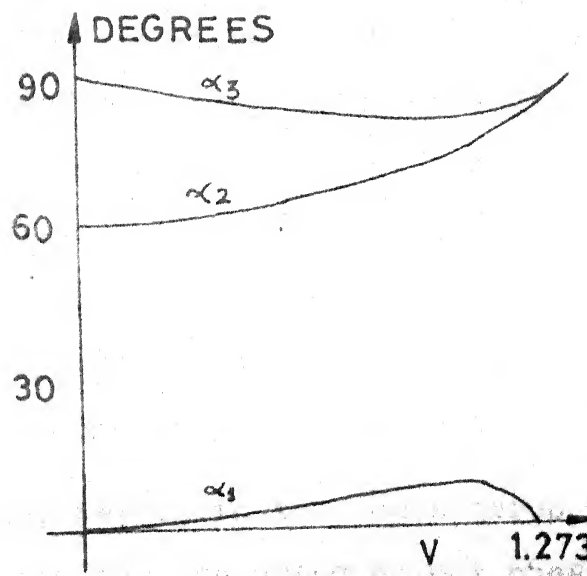


Fig 3.7b Variation of switching angle with modulation index

## CHAPTER 4

### MULTIMODE THREE PHASE PULSEWIDTH MODULATOR

The overall scheme of the pulsewidth modulator is presented. The digital implementation techniques for different modes of pulsewidth modulation are presented. The requirements for smooth transition between modes are elaborated. The software for implementing each mode and for smooth transition between modes is illustrated. Finally suggestions to improve the performance of the modulator are proposed.

#### 4.1 PWM INVERTER GATING SIGNALS:

A conventional three phase PWM inverter consists of six switches  $T_1$  to  $T_6$  as shown in Fig. 4.1(a). The commutation circuit has not been shown in the figure. The dc voltage supply can be considered as split into two sources with a fictitious ground. If the dc supply is obtained by rectification of three phase ac supply, then the fictitious ground corresponds to the neutral of three phase supply. The split source supplies equal voltages as shown in Fig. 4.1(b).

When any switch in the upper group conducts, the potential of the corresponding output phase becomes  $V_{dc}/2$ . Similarly when any switch in the lower group conducts, the

87541

potential of the corresponding output phase becomes  $-V_{dc}/2$ . The switches connected to the same phase should not conduct together to avoid a short circuit across the dc supply. Hence the phase output voltage can have only two states  $+V_{dc}/2$  and  $-V_{dc}/2$ .

During the operation of three phase PWM inverter, one of the switches connected to each phase is always ON. Hence at any state, three switches in the inverter are always ON. When phase A voltage goes from  $+V_{dc}/2$  to  $-V_{dc}/2$ , first the switch  $T_1$  is turned OFF and then  $T_4$  turned ON to avoid simultaneous conduction of  $T_1$  and  $T_4$  which will cause short circuit across dc bus known as 'shoot through fault'. Hence the gating signals for  $T_1$  and  $T_4$  are complementary to each other except for a small interval when both are zero.

The phase voltages are an amplified version of the gating pattern. The line to line voltage waveform can be determined from these by simple subtraction of the phase voltage waveforms.

The further discussions in this chapter deal with the generation of gating signal for each switch. The gating pattern for phase A will correspond for switch  $T_1$ , hence a High implies  $+V_{dc}/2$  at phase A output and Low implies  $-V_{dc}/2$

output. The gating pattern for  $T_4$  is complement of  $T_1$  gating pattern. The same is true for gating patterns of phase B and C and switches  $T_3$ ,  $T_6$  and  $T_5$ ,  $T_2$  respectively.

An illustrative figure to demonstrate this along with the line to line voltage waveform is given in Fig. 4.2.

#### 4.2 OVERALL SCHEME OF THE PULSEWIDTH MODULATOR:

The strategies adopted in the pulsewidth modulator in this thesis are described. The hardware and the software structure is also explained.

##### 4.2.1 Strategies Adopted in Pulsewidth Modulator:

For selection of strategies in the pulsewidth modulator, the merits and demerits of the pulsewidth modulation strategies were considered. Each pulsewidth modulation strategy shows characteristics desirable in certain range of output frequency. Hence a multimode strategy, which is an amalgamation of many distinct, desirable strategies is selected. The multimode strategy, with each mode distinctly differing from others, is easily implementable with the software manipulative capabilities of a microprocessor. The pulsewidth modulator designed follows the scheme in Ref.[14].

For a closed loop control of induction motor, independent control of voltage and frequency is desirable. For open

loop control of induction motor, the relation between voltage and frequency is linear except at very low frequencies where boosting of voltage is required and at frequencies above the rated value where the maximum rated voltage is maintained. The pulsewidth modulator designed in this thesis has such a  $V/F$  characteristics stored in a look up table. Independent control of voltage and frequency is also possible in the modulator implemented.

In the low output frequency range of 0 to 20 Hz the regularly sampled sinusoidal pulsewidth modulation is implemented. Above the SPWM range, optimal pulsewidth modulation strategy is selected. The optimal PWM smoothly transits through pulse dropping to square wave mode at the highest output voltage which occurs at rated output frequency of 50 Hz. The square wave mode continues till maximum output frequency of 100 Hz.

In the regularly sampled SPWM the most important design parameter is the ratio of carrier frequency to modulating frequency. The harmonic contents in the output voltage depends on this ratio. Larger the ratio, lower is the harmonic content and better is the output waveform. But as the ratio increases, the maximum carrier frequency increases for a moderate range of SPWM. With increase in carrier frequency

the number of commutations/second increase, increasing the commutation losses and requiring high speed commutation circuits. The maximum carrier frequency selected is around 370 Hz and the minimum is around 170 Hz. These are suited for thyristorised PWM inverters Ref.[9]. The maximum carrier frequency is attained at the modulating frequency of 10.5 Hz. The ratio is 36. It is desirable for this ratio to be a multiple of 3 to avoid generation of triplen harmonics in the three phase output voltage.

In the SPWM output frequency range of 0 to 20 Hz, asynchronous SPWM is implemented for 0 to 5 Hz, synchronous SPWM with ratio 36 for 5 to 10 Hz. For the output frequency range 10 to 20 Hz, gear-changing technique is adopted to implement synchronous SPWM with ratio 18.

The optimal PWM pattern selected has 3 notches per quarter cycle. The optimised tables have been taken from Refs. [15,21]. As the output voltage increases, the optimal patterns transits to pulse dropping mode and finally to square wave operation at maximum output voltage. Beyond this only the output frequency can be changed while the output voltage is constant. This is the constant power region in the induction motor - PWM inverter drive system.

The different modes, variation of output voltage, variation of carrier frequency with output frequency is shown in Fig. 4.3.

#### 4.2.2 Schematic of Hardware and Software Structure:

In this subsection the hardware and the software structure that has been implemented, are described. The explanation, the justification and the implementation of each PWM strategy is discussed in detail in the following sections.

For the generation of SPWM gating signals, the actual comparison via hardware between a triangular carrier wave and a sampled sinusoidal wave is not required. But with the help of the computational capabilities of a microprocessor, the pulsewidths can be computed. These are then converted into time delays by loading into the 'Pulsewidth Counter' which is clocked by a selectable clock frequency. Each phase requires an independent pulsewidth counter. Hence a three phase pulsewidth modulator requires three pulsewidth counters.

When a pulsewidth counter reaches terminal count, it generates an output signal which is used as an interrupt to the microprocessor. The three pulsewidth counters for the three phases have been implemented using the Intel 8253 timer.

The Intel 8253 programmable interval timer consists of three programmable 16-bit down counters. These are used as



Pulsewidth Counters for the three phases. All the three counters are programmed in mode  $\emptyset$  i.e. interrupt on terminal count. The outputs of these counters are connected to vectored interrupts of the 8085A microprocessors. Counter  $\emptyset$  is connected to RST 5.5, counter 1 to RST 6.5 and counter 2 to RST 7.5 interrupt.

The selectable clock frequency for the pulsewidth counters is obtained by programmable division of a higher frequency clock source. For a fine resolution in output frequency, the resolution in the clock frequency has to be much smaller. This is elaborated in detail in Sec. 4.3.1. This can be obtained by programmably dividing a very high frequency clock source by a large programmable count  $W_{tc}$ . The largest value of  $W_{tc}$  required for the pulsewidth modulator designed in this thesis is less than 12-bit in binary representation. The equations for determining the count  $W_{tc}$  are given in Sec. 4.3 and Sec. 4.5.

The high frequency clock for the programmable frequency, /programmable frequency divider should provide a divider,  $W_{tc}$ , is obtained from a 24 MHz crystal. The string of pulses of adjustable frequency to the pulsewidth counters. Hence it should generate an output pulse at terminal count, automatically reload and repeat again. This is accomplished by cascading counters and using load facility provided on most

4-bit counters. The 12-bit programmable divisor or count is latched and held at the input of the counters by two latches.

The Intel 8282 latches have been used. The counter is a synchronous UP counter 74LS161A with asynchronous clear. The standard cascade configuration with carry look ahead is implemented. The 74LS161A counters being UP counters, the value latched at the counter inputs by the 8282 latches has to be complemented of the calculated  $W_{tc}$ . The carry out (output) pulse from the cascaded counters has a pulsewidth smaller than the minimum required by Intel 8253. Hence it is stretched by a monostable before it is fed to Intel 8253 counters.

The output gating pattern is set at a part of 8255 by the microprocessor. This is isolated and amplified before it is sent to power switches in the inverter. The program for implementing the pulsewidth modulator is stored in EPROM's (2716).

The block schematic for the pulsewidth modulator is shown in Fig. 4.4.

The structure of the pulsewidth modulator is interrupt based and interrupt driven. Hence the software for most part consists of interrupt service subroutines for the three phases in different modes. The main program mostly deals

with keyboard inputs and CRT outputs, while the actual switching strategy of the pulsewidth modulator is delegated to interrupt service subroutines (ISS's).

The main program initialises ports counters and interrupts. It also initialises software pointers like sine pointers, read pointers, write pointers and software flag like change flag, mode flag etc. as described in the following sections. It selects an initial, default output frequency of 0.5 Hz, in the asynchronous mode frequency range. All computations for the asynchronous mode are accomplished, these are explained in Section 4.4. The proper clocking frequency to the pulsewidth counters is set and the first delay in a carrier is sent to pulsewidth counters. The switching strategy for the pulsewidth modulator has been initiated, now the entire switching operation is interrupt based and interrupt driven.

Having accomplished these tasks, the main program interacts with the user through keyboard and CRT display. The main program prompts the user to type C for change and A for abort. It is then in input mode expecting a command from the user.

Upon receiving change command, it asks and accepts new output frequency command. It identifies present mode from the mode flag, checks for step size between new output

frequency and present output frequency. It identifies new mode of operation. It completes computation of  $W_{tc}$  for new mode, gets new output voltage from V/F table. It also sets the mode and changes flag before displaying on screen the new mode of operation, output frequency and modulation index. For incorrect inputs appropriate error messages are displayed on CRT and the program returns to the input mode.

When an abort commands is received, the program disables the interrupts, clears the output trigger signals and by software reset returns to the host monitor program. The flow chart is shown in Fig. 4.5.

The requirements for the readpointers, writepointers, sine pointer flags like charge, mode etc. will be explained in the sections to follow. The computations required for the modes of pulsewidth modulator are explained in detail. The relevant equations for computing  $W_{tc}$  upon change in frequency command are also explained for each mode.

#### 4.3 IMPLEMENTATION OF SYNCHRONOUS SPWM:

The computational method for determining pulsewidth and generation of gating signals is explained for single phase and three phase synchronous SPWM. The minor changes required for the Gear Changing technique are also elaborated. The description of the software for implementing the synchronous SPWM is explained briefly.

#### 4.3.1 Single Phase Synchronous SPWM:

In a synchronous SPWM, the ratio of carrier frequency to modulating frequency is held constant. The variation of the output frequency also varies the carrier frequency, resulting in a constant number of output pulses in every cycle.

In a regularly sampled SPWM, the pulsewidth of one output pulse (Fig. 4.6) is given by

$$W_{tw} = \frac{T}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi + 1 \right) \text{ sec.} \quad (4.1)$$

where  $T$  : carrier period in seconds

$V_S$  : desired output voltage

$V_{SM}$  : maximum output voltage

$\phi$  : sampled sine value from sine table of unity amplitude.

There are different ways of generating the carrier interval  $T$  (Fig. 4.6). In the scheme implemented in this thesis,  $T$  is generated by dividing a variable frequency clock by a constant number  $N$ . Hence a carrier interval corresponds to  $N$  clock periods. With the variation in clock frequency, the carrier period changes. The output pulsewidth is then given by eqn. (4.2)

$$W_{tw} = \frac{N}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi + 1 \right) \quad (4.2)$$

where  $W_{tw}$  : output pulsewidth in units of clock period

Hence in a carrier interval of  $N$  clock periods, the output trigger pulse is high for  $W_{tw}$  clock periods and low for the remaining  $(N - W_{tw})$  clock periods. The gating pulse for the complementary power switch is the complement of above mentioned trigger pulse. An important feature of the symmetric regularly sampled SPWM is that the output pulse is symmetrically placed within a carrier interval. Hence with

$$W_{td} = \frac{1}{2} (N - W_{tw}), \quad (4.3)$$

the sequence of delays in a carrier interval is  $W_{td}$ ,  $W_{tw}$ ,  $W_{td}$  as shown in Fig. 4.6. The output trigger pulse is set high for  $W_{tw}$  period and low for the remaining clock periods. This is shown in Fig. 4.6.

The pulsewidth counts  $W_{td}$ ,  $W_{tw}$  are converted into correct delays loading into a 'Pulsewidth Counter', clocked by selectable frequency clock.

As has been explained earlier, in a synchronous SPWM, the carrier frequency changes with modulating frequency. But a carrier interval consists of  $N$  clock periods of the clock fed to the pulsewidth counter. Hence with changes in the clock frequency to the pulsewidth counter, the carrier interval and hence the output frequency of the PWM output waveform changes.

The variable frequency clock for the pulsewidth counter is generated by dividing a higher frequency clock by a programmable word ' $W_{tc}$ ' (Fig. 4.4). The word  $W_{tc}$  is naturally dependent on the output frequency,  $N$  and also number of pulses in one output cycle of the PWM waveform. It is given by eqn. (4.4)

$$W_{tc} = \left( \frac{F_{s \max}}{f_{s \max}} \right) \cdot \frac{f_1}{n \cdot F_s} \cdot \frac{1}{N} \quad (4.4)$$

where  $F_s$  : digital representation of output frequency

$F_{s \max}$  : digital representation of maximum output frequency

$f_1$  : high frequency clock source (24 MHz)

$n$  : carrier frequency/output frequency (36)

$N$  : Number of clock pulses/carrier (256)

$f_{s \max}$  : maximum output frequency (100 Hz)

The values written in brackets are selected for implementation in this thesis. An example to illustrate the above equation is given in Fig. 4.18,

For a fine resolution in the absolute value of the output frequency, the carrier interval resolution has to be better by ' $n$ ' ( $f_c/f_m$  ratio). The carrier interval itself is generated by  $N$  clock periods of pulsewidth counter. Hence extremely fine resolution in the clock frequency of pulsewidth counters is essential.

The frequency of clock to pulsewidth counter ' $f_{\text{clock}}$ ' is given by eqn. (4.5).

$$f_{\text{clock}} = \frac{f_1}{W_{tc}} \quad (4.5)$$

$f_{\text{clock}}$  : clock frequency to pulsewidth counter

To obtain fine resolution in  $f_{\text{clock}}$ , fine variations in  $W_{tc}$  should be possible. But  $W_{tc}$  can change only in order of natural numbers, hence  $W_{tc}$  itself should be large. Hence from eqn. (4.4) it can be deciphered that  $f_1$  has to be very large.

Hence the method for calculating pulsewidths, generating delays is explained above. But how is the correct sequence maintained how is the output trigger pulse generated within a carrier interval is explained below.

When a pulsewidth counter counts down any interval  $W_{tw}$  or  $W_{td}$  and reaches the terminal count, it generates an interrupt to the microprocessor. The microprocessor correctly outputs the next pulsewidth count to the pulsewidth counter. Thus there are three interrupts in a carrier interval Fig. 4.6.

For generation of output trigger pulses, hardwired techniques can be adopted. In this thesis, the microprocessor upon getting a delay interrupt, identifies state within a



carrier period, sends next pulsewidth count to the counter and appropriately sets an output port of 8255. This is explained below in detail.

When the pulsewidth counter is counting down a  $W_{td}$  count, the trigger pulse - output set is '0' or low Fig.

4.6. When the pulsewidth counter is counting down  $W_{tw}$  count, the output set is high or '1'.

Hence we have seen how the pulsewidth counts for a carrier cycle (implying one sine sample) are determined, how the counts are generated into delays, how the proper sequence is maintained and finally how the output trigger pulses generated in a carrier interval. Finally how are the calculations for sequential carrier cycles are carried out and how these results in the generation of trigger pulses for generating ac output voltage at PWM inverter output is explained below.

When a carrier interval is completed, the modulating sine wave is sampled to get next sample  $\phi$ . Since only 'n' equally displaced sine samples of the modulating wave are required, it can be kept in a lookup table. The effect of different output voltages is taken into eqn. (4.2), hence only 'n' sine samples of a sine wave of unit amplitude are stored in a sine table.

Based upon the new sine sample  $\phi$ , the computation of new pulsewidth counts  $W_{tw}$ ,  $W_{td}$  is completed. To avoid time delay due to computation of  $W_{tw}$ ,  $W_{td}$ , the pulsewidth counts are determined one carrier earlier. That is, even as the pulsewidth counter is counting down, the computations for the pulsewidth counts required in the next carrier i.e. based on next sine sample is commenced. The computations are completed before the next carrier cycle begins. During this computation, when a pulsewidth delay interrupt is caused, the microprocessor goes to the interrupt service routine where it sends next pulsewidth in the carrier to the counter, sets output at port and returns to computation routine. Hence at the beginning of a carrier cycle, the pulsewidth counts for the carrier cycle are already determined and available. Thus there is no delay time due to the computations.

In steady state operation, this sequence is repeated, the only difference between consecutive carrier cycles is the sine sample  $\phi$ , which is retrieved from sine table using a sine pointer. Hence the sinepointer is a modulo- $n$  software counter. The state of output voltage waveform is depicted by the value of sine pointer. With a complete scan of the sine table, ' $n$ ' carrier cycles are generated, resulting in ' $n$ ' output trigger pulses. These will result in an ac cycle at the output of the inverter.

#### 4.3.2 Implementation of Three Phase Synchronous SPWM:

The explanation for three phase synchronous SPWM assumes the description of single phase synchronous SPWM as the two are similar. Only the salient and distinguishing points for a three phase system are given below.

In a three phase system, the voltages of the three phases are shifted from each other by 120 degrees. Hence the gating pattern generated by a three phase pulsewidth modulator for the three phases should also have this phase shift. In the computational technique explained in Sec. 4.3.1, the state of the output phase voltage is determined by the value of the phase sine pointer. Hence to incorporate the 120 degree phase shift the individual phase sine pointers have to be appropriately set. An example of this is given in Sec. 4.3.3. At the end of every carrier interval all the sine pointers are changed by one, hence the phase shift is maintained.

The computation of the pulsewidth count for the three phases are identical except that each phase retrieves its sine sample using its own sine pointer. The relevant equations are listed below.

$$\begin{aligned}
W_{twA} &= \frac{N}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi_A + 1 \right) & W_{tdA} &= \frac{1}{2}(N - W_{twA}) \\
W_{twB} &= \frac{N}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi_B + 1 \right) & W_{tdB} &= \frac{1}{2}(N - W_{twB}) \\
W_{twC} &= \frac{N}{2} \left( \frac{V_S}{V_{SM}} \cdot \phi_C + 1 \right) & W_{tdC} &= \frac{1}{2}(N - W_{twC})
\end{aligned} \tag{4.6}$$

These pulsewidth counts have to be converted into delays by loading into independent pulsewidth counters clocked by a common selectable frequency clock. Hence three pulsewidth counters are required, one for each phase

The sequence of loading pulsewidth count into a pulsewidth counter in a carrier interval is  $W_{td}$ ,  $W_{tw}$ ,  $W_{td}$  for a phase e.g.  $W_{tdA}$ ,  $W_{twA}$ ,  $W_{tdA}$  for phase A pulsewidth counter. When a pulsewidth counter reaches terminal count it causes an interrupt to the microprocessor. The microprocessor jumps to the phase interrupt service subroutine where it sends the next sequential pulsewidth count to the pulsewidth counter and sets correct bits at the output port appropriately. Each phase requires two bits at the output to set the output trigger pulses for the complementary switches connected to each phase. When the pulsewidth counter is counting  $W_{tw}$  delay for a phase,

the output trigger for the phase is set high or '1'. When the pulsewidth counter is counting  $W_{td}$  delay for a phase, the output trigger for the phase is set low or '0'.

Thus in a three phase pulsewidth modulator, there are three interrupting sources, three interrupt service subroutines, three individual phase pulsewidth counters and six bits at the output port which have to be set or reset.

In a carrier cycle, each phase pulsewidth counter generates three interrupts after counting three pulsewidths. But the carrier interval interrupt of all the phases coincide as shown in Fig. 4.7. Hence instead of allowing all the three pulsewidth counters from causing interrupts, only one is allowed to cause an interrupt and recognised. Hence in a carrier interval there are 7 interrupts. In the software implemented in this thesis, phase A is allowed to cause a Carrier Interval Interrupt and is termed the 'Master Phase'. With increased hardware burdens the number of interrupts per carrier can be further reduced Ref. [21].

At a carrier interrupt, the Master Phase has to send  $W_{tdA}$ ,  $W_{tdB}$ ,  $W_{tdC}$  pulsewidth counts based on next sine samples, to their respective pulsewidth counters. A brief explanation of the method determining the pulsewidth counts for the three phases is given below.

At a pulsewidth delay interrupt, the next pulsewidth count to be sent to the pulsewidth counter must be available to avoid time delay in computation of the next pulsewidth count. Hence all the six pulsewidth counts for the three phases are carried out one carrier cycle in advance. Even as the phase pulsewidth counters are generating delays and causing interrupts, the microprocessor in the carrier ISS (Fig. 4.9) completes the computation of the pulse width counts required for next carrier interval. If during the computation of pulsewidths for the three phases, an interrupt is caused, the microprocessor immediately jumps to the interrupt service subroutine for the interrupting phase, outputs next pulsewidth count, sets output appropriately and returns to complete the computations. This eliminates the time delay due to computation of pulsewidth counts.

From one carrier to the other, the computations are carried out in a similar manner and only the sine samples used for the computations change. The sine sample used for computing pulsewidth counts for the next carrier interval is one displaced from the sine sample used for computing pulsewidth counts presently being output to the pulsewidth counters. Hence carrier cycle after carrier cycle the sine pointers for all the phases are corrected. Hence for 'n' sine samples in

a sine lookup table (same as  $f_c/f_m$  ratio) the phase sine pointers are modulo- $n$  software counters. The computed pulsewidth counts for all phases are stored in scratch memory, these are appropriately retrieved and sent out by the interrupt service subroutines for the three phases.

#### 4.3.3 Software Description for Synchronous SPWM:

The software logic for the synchronous SPWM switching strategy of the three phase pulsewidth modulator is implemented entirely in the interrupt service subroutine for the three phases.

The master phase A has three interrupts in a carrier interval. The first interrupt is caused after  $W_{tdA}$  delay Fig. 4.7, the second after an additional  $W_{twA}$  delay and finally the carrier interrupt after an additional  $W_{tdA}$  delay. The slave phase B has only two interrupts in a carrier interval, first after  $W_{tdB}$  delay and the second after an additional  $W_{twB}$  delay. The same is true for phase C where the two interrupts are caused after  $W_{tdC}$  and  $W_{twC}$  delays respectively. The software logic to be accomplished at each of these interrupts for an individual phase is slightly different e.g. upon first interrupt in say phase B, the ISS for phase B should send  $W_{twB}$  pulsewidth count to pulsewidth counter for phase B and set the output trigger bit for phase B at output port high or '1'.

But upon the second interrupt it should just set output trigger signal low for phase B. It need not send  $W_{tdB}$  to pulsewidth counter as only master phase A causes carrier interrupt.

To improve speed of execution of program, separate interrupt service subroutines have been written to accomplish the logic on first, second and carrier interrupt (present only in master phase) of a carrier interval. The sequence of interrupts is fixed as first, second, carrier (present only for master phase) and then back to first interrupt logic in next carrier interval. The individual interrupt service subroutines after accomplishing the required task for the interrupt, place the address of the ISS for the succeeding interrupt at an interrupt address pointer for the phase. Hence phase A ISS for the first interrupt places the address of the ISS for the second interrupt at phase A interrupt address pointer. The phase A ISS for the second interrupt points to the carrier ISS and the carrier ISS points to ISS for the first interrupt before returning. For phases B and C there is no carrier interrupt, hence the ISS for second interrupt points to ISS for first interrupt before returning.

The first interrupt in a carrier interval is caused after  $W_{td}$  delay for the phase. Hence the phase ISS for first interrupt sets the output high, loads the phase pulsewidth



counter with  $W_{tw}$  delay computed for it. Before returning from ISS, it points to the ISS for the second interrupt by placing its address at the interrupt address pointer for the phase. The above sequence is followed by routines of the first interrupts for all the phases.

The second interrupt in a carrier interval is caused after additional  $W_{tw}$  delay for the phase. Hence the phase interrupt service subroutine for second interrupt sets the output for the phase low. This part is common to ISS for second interrupt for all phases. In the slave phases B and C, the address of first interrupt ISS is placed at the respective interrupt address pointers before returning. In master phase i.e. phase A ISS for second interrupt, the address of the carrier ISS is placed at the interrupt address pointer before returning. The first and second interrupt ISS for all phases for all modes of SPWM are identical. A flow chart for these is given in Fig. 4.8. The major part of software logic for a mode is implemented in the carrier interrupt ISS. Hence in the explanation for other modes of SPWM only the carrier ISS will be explained.

In the carrier interrupt service subroutine for synchronous mode, the change flag is tested. The explanation for change flag is given in Section 4.6. If reset, the program

jumps to logic part of software implementation of synchronous mode (Fig. 4.9). Here it outputs  $W_{tdA}$ ,  $W_{tdB}$ ,  $W_{tdC}$  to their respective pulsewidth counters. It then points interrupt address pointer master phase to the ISS for first interrupt, enables interrupts and proceeds to computation of pulsewidth counts for next carrier interval. For computing pulsewidth counts, it gets sinepointers for individual phases, retrieves sine samples  $\phi_A$ ,  $\phi_B$ ,  $\phi_C$  from sine lookup table, decrements sine pointers (as the sine table stores inverted sine table, refer Sec. 4.7.3) stores them back. Then proceeds to computation of pulsewidth counts as given by eqn. (4.6). It then returns to the background routine.

If the change flag is set, it is tested for  $\phi 2H$  if found true all sinepointers are made even. The following part is executed if change flag is set to  $\phi 1H$  or  $\phi 2H$ . The complemented  $W_{tc}$  is sent to the 8282 latches, the modulation index changed by new modulation index. Then change flag is reset to  $\phi 0H$ . The address of new ISS placed at interrupt address pointer for master phase. The program then jumps to logic part in new ISS. The complete flowchart is given in Fig. 4.9.

#### 4.2.4 Gear Changing Technique Implementation:

In the implementation of regular SPWM in the output frequency range of 10 Hz to 20 Hz, gear changing technique

is used. The difference and implementation technique for this is explained.

To reduce the commutations/second at higher output frequency, the ratio of carrier frequency to modulating (output) frequency ( $n = f_c/f_m$ ) is reduced (Fig. 4.3). Reduction by half is preferred. The basic structure of hardware and software logic remains same, only the computational logic changes slightly.

The number  $W_{tc}$  for programmable frequency divider to generate the clocking frequency for pulsewidth counters is now given by eqn. (4.7).

$$W_{tc} = \frac{F_{S \max}}{f_{S \max}} \cdot \frac{f_1}{n' \cdot F_S} \cdot \frac{1}{N} \quad (4.7)$$

where  $n' = \text{new ratio } f_c/f_m = n/2$

Also the operation of modulator in a carrier is identical to that explained above but alternate sine samples are used to compute pulsewidth counts for next carrier. Hence in one scan of sine table,  $n/2$  sine samples are used for computing pulsewidths,  $n/2$  carriers are generated resulting in  $n/2$  output pulses every ac output cycle.

The logic part of carrier ISS for synchronous mode with ratio 18 is identical to synchronous mode with ratio 36,

except sine pointers being decremented twice every carrier interval. The logic for mode changing is different and explained briefly here (for details refer to Sec. 4.6).

If change flag is 01H implying change within synchronous mode range. The new  $W_{tc}$  complement is sent to 8282 latches, new modulation index replaces old modulation index. The change flag is reset. The flow chart is given in Fig. 4.10.

If change flag is 02H, it implies new mode is optimal PWM. The sine pointer for phase A is tested to check if it is at any 60 degree point namely 0, 60, 120, 180, 240 or 300 degrees. When found true, the interrupt numbers for the three phases required in optimal PWM ISS are properly set, complemented  $W_{tc}$  sent to 8282 latches, modulation index changed. The address of optimal ISS's for all three phases placed at respective interrupt address pointer. The program then jumps to 'marker' in master phase A ISS.

#### 4.4 IMPLEMENTATION OF ASYNCHRONOUS SPWM:

The computational technique for determining the pulse-width counts in a carrier interval of asynchronous mode is same as that for synchronous mode already explained in Section 4.3. The technique for sampling the sine table is different and is explained below.

#### 4.4.1 Implementation of Single Phase Asynchronous SPWM:

In a asynchronous SPWM, the frequency of the clock of pulsewidth counter is held constant at the maximum permissible value. This results in the maximum carrier frequency as a carrier interval is  $N$  clock periods of pulsewidth counter clock.

With changes in output frequency, the number of carrier intervals in an output ac cycle varies hence  $f_c/f_m$  is not constant and need not be an integer. To generate one output cycle, there are ' $n$ ' sine samples in the sine lookup table. Hence the number of carriers per sine sample is not fixed. This is unlike synchronous SPWM where one carrier interval is generated for one sine sample. Hence in asynchronous SPWM, along with the carrier interval an additional 'Sampling Interval' has to be generated. One sine sample is sampled in every sampling interval. There are two techniques of implementing this sampling interval, the hardware technique and the software technique.

In the hardware technique, the sampling interval is generated independently by a high frequency clock (can be same as that used for carrier interval generation) and a programmable frequency divider counter of large acceptable counts. The output of this counter gives points of completion

of the sampling interval and interrupts the microprocessor. The programmable divide by count  $W_{TC}$  can be determined from eqn. (4.8)

$$W_{TC} = \left( \frac{F_{S \max}}{f_{S \max}} \right) \cdot \frac{f_{in}}{n \cdot F_S} \quad (4.8)$$

where  $f_{in}$ : frequency of clock source

$n$  : number of sine samples in lookup table.

The sampling interval interrupt informs the processor that the sampling interval is complete and delay for new sine samples already computed can be sent to pulsewidth counter. The processor gets next sine samples for all phases and computes pulsewidth counts for next sampling interval.

In the software technique, the number of constant frequency clock pulses to the pulsewidth counter within a sampling interval is determined from eqn. (4.9).

$$S = \left( \frac{F_{S \max}}{f_{S \max}} \right) \cdot \frac{f_{C \max}}{F_S} \cdot \frac{N}{n} \quad (4.9)$$

where  $f_{C \max}$ : maximum carrier frequency and other are defined in eqn. (4.4)

At every carrier interrupt, the number of clock pulses remaining within the present sampling interval is determined by subtracting  $N$  from the previous remainder resulting in the present

remainder. When the present remainder becomes zero or less than zero the sampling interval is completed. New sampling interval is started by sending pulsewidth counts to pulsewidth counters based on new sine samples. The excess clock pulses allowed in the previous sampling interval is subtracted from 'S' to get new remainder. The computation of pulsewidth counts required for next sampling interval is commenced. Thus integral carrier cycles are allowed in a sampling interval.

The merits of the software scheme are reduced hardware, easy implementation. For these reasons and also due to shortage of interrupt lines available on the host system the software technique is used. There is a possibility of the introduction of 'Beats' due to integral carrier cycles in both the techniques.

The logic of computing pulsewidth counts, converting them into delays by loading counter. Getting interrupted at correct delays and setting output etc. as explained in Section 4.3.1 is same for single phase asynchronous SPWM. The only difference is the determination of the sampling interval as explained above.

#### 4.4.2 Implementation of Three Phase Asynchronous SPWM:

As explained in Sec. 4.4.1, the principle of generation of output trigger pulses in a carrier interval using computational

technique also holds true for asynchronous mode of SPWM. It is just the instant or method of sampling which is different. The two techniques for generation of sampling interval have been explained in Sec. 4.4.1. In the implementation of asynchronous mode SPWM in this thesis, the software technique is used.

In a three phase pulsewidth modulator, there are seven interrupts in a carrier interval for all phases. Three first interrupts for the three phases, three second interrupts and only one master phase interrupt. It is in the ISS of the carrier interrupt of the master phase that the software technique of determining sample interval is implemented. The rest of operation and reasoning is already given in Sec. 4.3.2 and Sec. 4.4.1.

#### 4.4.3 Software Description of Asynchronous Mode SPWM:

The ISS for phases B and C just send the next pulsewidth count to the respective pulsewidth counter and set the output trigger pulse for respective phases appropriately. They do not play any major part in the sampling technique or switching strategy. Hence these are identical as required for synchronous SPWM. Infact for the entire SPWM range, the ISS for phases B and C do not change and the same are retained even on mode change within SPWM range. The logic for the first and



the second interrupt ISS for the master phase also does not change, only in second interrupt ISS, the address of carrier interrupt ISS for asynchronous mode is placed at phase A interrupt address pointer. The flow charts for these are given in Fig. 4.8(a), Fig. 4.8(b).

It is in the carrier ISS of the master phase that the entire logic of sampling interval determination and computation of new pulsewidth counts is implemented. In the carrier ISS of the asynchronous mode, a check is made if the present sampling interval is completed. This is accomplished by subtracting  $N$  (256 in the implemented scheme) from the present remainder of clock pulses in the sampling interval. If the new remainder becomes equal to or less than zero, it is inferred that the present sampling interval has been completed. Otherwise the old sampling interval continues. Hence read pointers are corrected to retrieve and output old pulsewidth counts  $W_{tdA}$ ,  $W_{tdB}$ ,  $W_{tdC}$  to their respective pulsewidth counters. It then points the interrupt address pointer to ISS for first interrupt in master phase and returns.

For a new sampling interval, the change flag is polled (details of mode changing requirements are given in Sec. 4.6). If set, complemented  $W_{tc}$  is sent to 8282 latches, new modulation index replaces old modulation index, new 'S' (eqn. (4.9)) is

placed at remainder if required, then the interrupt address pointer for master phase A is pointed to carrier ISS of new mode, change flag reset and the program jumps to logic port of new ISS.

If change flag is not set, then the extra clock pulses allowed in the previous sampling interval is deducted from 'S' and becomes remainder. The new  $W_{tdA}$ ,  $W_{tdB}$ ,  $W_{tdC}$  pulsewidth counts already computed and stored in scratch pad are sent to respective phase counters and address of first interrupt ISS placed at interrupt address pointer of master phase A. Interrupts are enabled before proceeding to computing pulsewidth counts for the three phases based on next sine samples. These will be required in the next or succeeding sampling interval. With this the ISS returns to main program. The flowchart is shown in Fig. 4.11.

#### 4.5 IMPLEMENTATION OF OPTIMAL PWM:

The optimal PWM has been used for the output frequency range of 20 to 50 Hz in this thesis. The optimal trigger pulses and hence output waveform smoothly transits through pulse dropping to square wave mode without any transient overvoltages. The optimal waveform selected has 3 notches/quarter cycle (Fig. 4.12).

Optimal PWM is a lookup table and pattern retrieval technique, which is suited for microprocessor implementation, Ref.[14,15,16]. The lookup table consists of notch counts which are converted into desirable delays by sending to a 'Notch Counter' with selectable clock frequency. (In physical reality the notch counters and the pulsewidth counters referred in Sec. 4.3, Sec. 4.4 are the same counters of 8253, only for explanation they are referred by the above names). The output trigger pulse to be set during the counting of any notch count is called pattern. It is shown as 'Trigger Pulse' in Fig. 4.2. In determining the notch counts and pattern for a optimal waveform, the number of switchings/quarter cycle are decided. Then a certain performance index is selected. Optimisation of the pattern is done for each output voltage value, resulting in the determination of the notch angles for each output voltage value. These notch angles are suitably, linearly translated into notch counts. Depending upon the implementation logic and different distinct nonrepetitive notch widths, a minimum number of notch counts are required to define and generate the optimal PWM delays. The counts are stored in the lookup table. The state of output during generation of each delay is also stored in pattern lookup table. Thus within a lookup table the pointer for selecting proper notch count is the output voltage value or modulation

index. The changes in output frequency are effected by changing the clock frequency to 'Notch Counter'. The changes in clock frequency to notch counter are accomplished by programmably dividing a higher frequency clock source  $f_1$  by  $W_{tc}$  (Fig. 4.4). For a linear translation of 'N' count corresponding to 60 degree interval in the output waveform, the count  $W_{tc}$  is given by eqn. (4.10)

$$W_{tc} = \left( \frac{F_{S \max}}{f_{S \max}} \right) \frac{f_1}{6 \cdot F_S} \cdot \frac{1}{N} \quad (4.10)$$

where N is number of clocks per 60 degree interval and other quantities are defined in eqn. (4.4).

Thus for a change in output voltage, the notch counts accessed change but not clock frequency. For change in the output frequency alone, say increase the notch counts remain same but the clock frequency increases to reduce the actual delays generated. This results in faster repetition of output delays. The output trigger pulse corresponding to each notch count has to be accessed and set. This basically depends on the implementation technique. The method used for implementing this is explained later.

#### 4.5.1 Implementation of Single Phase Optimal PWM:

The optimal trigger waveform for an entire cycle has been shown in Fig. 4.12. It can be seen from the quarter wave

symmetry in the waveform that only four notch counts  $W_1, W_2, W_3, W_4$  are essential to define and reproduce an output optimal PWM trigger cycle. Thus in one output trigger cycle, these notch counts are accessed in the sequence  $W_1, W_2, W_3, W_4, W_3, W_2, W_1, W_1, W_2, W_3, W_4, W_3, W_2, W_1$ . The state of output trigger pulse for these notch widths is 0,1,0,1,0,1,0,1,0,1,0,1,0,1 respectively. Even though these are just toggling they are stored in pattern lookup table. The output trigger pulse for the complementary switch is complement of the above pattern. The number of output states stored in the pattern table can be reduced with more software manipulations. But the table itself being so small, it is usually entirely stored.

Now during the operation of pulsewidth modulator, depending on the output frequency command, the proper clock frequency for the notch counter is selected. Depending on the output voltage requirement the pointer for accessing the notch counts is constructed. Then an interrupt number software counter is initialised and first notch count say  $W_1$  is accessed using interrupt number and is sent to notch counter. Even as the notch counter is being decremented to generate the notch delay, the output trigger pulse is accessed using interrupt number from the pattern table and sent to output port. When the notch counter reaches terminal count it causes an interrupt to the microprocessor. The microprocessor

accesses next notch count in optimal waveform and sends it to the notch counter, it corrects the interrupt number software counter, accesses new output pattern and sets it at the output port. Thus in one cycle of trigger pulses, fourteen interrupts are caused, the fourteen states of pattern table are accessed and set at output port. Thus the interrupt number software counter is a modulo-14 counter. It can also be seen from Fig. 4.12, in one output cycle the delay  $W_1$  to  $W_3$  are accessed four times and delay  $W_4$  twice. Hence only the interrupt number uniquely defines the state of output trigger waveform. The interrupt numbers at different states of output trigger waveform are shown in Fig. 4.12.

As the modulation index increases, the notch width counts  $W_1$ ,  $W_3$ ,  $W_4$  (Fig. 4.12) reduce. When these become small so that the delays generated by it would be below the minimum acceptable to the power circuit, they are dropped from the optimal waveform. The counts for notch width in new optimal waveform is determined and stored in notch count table, an entry of zero is made in table of notch counts for the notch (delay) dropped. Thus before a new notch count is sent to notch counter, a check should be made to ensure it is non-zero. If zero, the interrupt number should be corrected and next notch count should be accessed. This process should continue till a non-zero notch count is accessed. This ensures smooth

transition from optimal PWM through pulse dropping optimal waveform to finally square wave mode. In square wave mode only  $W_2$  will have non-zero notch count.

By repetitive cyclic accessing of notch counts, sending to notch counter, accessing output state from pattern table and setting at output port at successive notch delay interrupts the optimal PWM trigger signal is generated at output port.

#### 4.5.2 Implementation of Three Phase Optimal PWM:

The difficulties in implementing three phase optimal PWM are explained. Scheme adopted in this thesis to overcome the problems is presented.

For the generation of three phase PWM trigger pulses, independent operation of all the three phases i.e. each phase behaves as though it is a single phase PWM is not easily feasible. The problem of operating them as three independent phases are as follows. (1) It is difficult to ensure the maintenance of proper phase relationship between the output phases. The phases may slide with respect to each other due to pending interrupts. (2) Problems are faced during mode changes. To start the three phase scheme of independent individual phases, additional notch count tables of large sizes are required to have efficient and quick mode changing.

In the implemented scheme, the optimal PWM waveform is subdivided into 6 equal parts each of 60 degree interval as shown in Fig. 4.13. The points for dividing into smaller intervals are 0, 60, 120, 180, 240, 300 degree points of the output voltage waveform, they will hence-forth be called 'Markers'.

The basic operation of an individual phase in the three phase pulsewidth modulator remains same as explained in Sec. 4.5.1. That is notch count is sent to notch counter, the output trigger state is retrieved from pattern lookup table using interrupt number. When the notch counter reaches terminal count it causes an interrupt to the microprocessor. The microprocessor jumps to ISS where next notch count is sent to notch counter, next output trigger state set, interrupt counter decremented etc. In a three phase pulsewidth modulator, three such notch counters are required, resulting in three interrupting sources, three interrupt number counters are required. The delay tables and pattern table are accessed by phase interrupt numbers appropriately. The basic change comes in the proper initialisation and maintenance of these phase shifts. The technique adopted in this thesis to ensure these is explained below.

In the implemented scheme, the optimal PWM trigger pattern and hence waveform is subdivided into 6 equal parts



each of 60 degree as shown in Fig. 4.13. In the adopted scheme, only one phase, the master phase, i.e. phase A, is allowed to cause marker interrupts. Other phase ISS stop sending delay to their pulsewidth counters one delay (notch/transition) before a marker but set the output state or pattern for the respective phase at correct pins of the output port (six output pins are set, two by each phase ISS). Upon a marker interrupt, the master phase retrieves correct notch count for each phase using the respective interrupt numbers. These are sent to the respective notch counters. The output trigger state for each phase is retrieved from the pattern table using again phase interrupt numbers. These are set at output port pins appropriately and the phase interrupt numbers are corrected. Hence the markers ensure that no sliding of phases is allowed, once the phase relationship is initialised, it is properly maintained.

A look at the optimal PWM trigger pattern with markers inserted, Fig. 4.14, shows a quarter wave symmetry. Five notch counts  $W_1, W_2, W_3, W_4, W_5$  are sufficient to generate the notch widths in the optimal PWM waveform. The state of the trigger pulse during the generation of each of the notch delays is stored in pattern lookup table.

The output trigger pulse waveform is generated by accessing notch counts, generating delays and setting output

implies 120 degree point of phase B output waveform and interrupt number phase C = 013H implies 240 degree of phase C output waveform. Thus by initialising the individual phase interrupts for phase A,B,C, as 12H, 07H, 0BH respectively, the proper phases shifts are initialised. There are five other combinations to initialise the phase shifts. All these occur at marker interrupts and can be seen in Fig. 4.14. Thus initialisation and maintenance of phase relationship is achieved with the master phase concept.

As output voltage increases, the few notch counts reduce. When the notch width generated by loading into notch counter is below the minimum acceptable output pulsewidth, these transitions in optimal pattern are dropped, an entry of zero is made in notch count table for dropped notch. The notch count for new optimal waveform is stored in the other notch count locations. During the operation of three phase pulsewidth modulator, before a notch count is sent to counter, it is checked for zero value. If notch count is not zero it is sent to counter, output trigger signal set and phase interrupt number decremented. If zero, the operation for master phase A and for other phases B and C slightly differ. In master phase A the next non-zero delay within marker is accessed after correcting interrupt number for phase A. This is sent

to notch counter, new output trigger state is set and interrupt number decremented. For phases B and C the next non-zero notch count within a marker is accessed. If it is one before a marker then only output trigger state is set and phase interrupt number decremented. Otherwise the notch count is sent to counter, output trigger state is set and interrupt number decremented. With this a smooth transition occurs from optimal PWM waveform through pulse dropping to finally square wave mode of operation. In square wave mode only  $W_2$ ,  $W_3$  have non-zero notch count. Thus smooth transition to square wave mode is achieved.

#### 4.5.3 Software Description of Optimal Mode PWM:

The software logic for implementing the optimal PWM is also entirely in the interrupt service subroutines for the three phases. Phase A has been selected as Master Phase and has to do certain extra functions. The ISS for phases B and C are identical.

The optimal PWM is a lookup table (contains notch counts) and pattern (output trigger state) retrieval scheme. As has been explained in Section 4.5.2, the optimal notch counts have to be accessed in a sequence  $W_1, W_2, W_3, W_4, W_5, W_4, W_3, W_2, W_1, W_1, W_2, W_3, W_4, W_5, W_4, W_3, W_2, W_1$  (refer Fig. 4.14). The least significant byte of notch count pointer is the desired output

voltage or modulation index. The most significant byte depends on the state of the output phases, reflected by interrupt numbers for each phase. The required output trigger at each interrupt is retrieved using phase interrupt number as pointer. For ease and simplicity of programming, the relationship between most significant byte of pointer to notch count tables and the phase interrupt numbers has been stored in a lookup table. The notch counts  $W_1$  to  $W_5$  are stored in memory with most significant byte as  $\text{0F5}$  to  $\text{0F9H}$ . These are required to interpret the contents in the lookup table. The contents of the lookup table is given in Table 2. An additional meaning is adopted for Bit 7 and Bit 6 of the most significant byte of notch count pointer, these are explained briefly. Bit 7 =  $\emptyset$  of the most significant byte to notch count pointer implies that the present interrupt is a marker interrupt. This is used by master phase A. Bit 6 =  $\emptyset$  for most significant byte of notch count pointer implies that present interrupt is one preceding a marker interrupt. This is used by phase B and C ISS. In light of the above explanation and Fig. 4.14 the contents of interrupt number versus most significant byte of notch count pointer become very clear. The above table is used by all the phase ISS to retrieve the most significant byte of notch count table pointer.

For phase B and C, the marker interrupts are not caused hence sending notch count to counter is stopped one transition before a marker interrupt will occur. This is determined from bit 6 of the most significant byte of notch count pointer as explained earlier. But the output trigger state is obtained from pattern table using interrupt number as pointer and set at correct output pins for the phase, then the interrupt number is decremented. Explanation of a typical slave phase ISS is given below.

Upon an interrupt for slave phase, the microprocessor jumps to the ISS of interrupted phase. In the ISS, the phase interrupt number is used to access the most significant byte of notch count pointer. Bit 6 of most significant byte of notch count pointer is checked for zero. If zero, the next output state for the phase is retrieved from pattern table using phase interrupt number. The phase interrupt number is then decremented, the phase notch counter is disabled and ISS returns after enabling interrupts. If Bit 6 is set to 1, the notch count pointer is constructed with this value and least significant byte as output voltage. This is used to access next notch count, this is sent to phase notch counter. The output trigger signal is retrieved from pattern table and set appropriate bits of the output port. The phase interrupt number is decremented modulo-18, it then enables interrupts.

The program then checks further notch counts within marker and adjusts the interrupt number to access first non-zero notch count within next marker on succeeding interrupt. This is useful to avoid time delays due to this checking if it is done after a interrupt is recognised. The flow chart for a typical slave phase is shown in Fig. 4.16.

The optimal PWM master phase A causes all 60 degree 'marker' interrupts in addition to its own required interrupts for changing output trigger pulse. Upon a master phase interrupt, the microprocessor jumps to master phase ISS. In the master phase ISS, the interrupt number of phase A is used to access most significant byte of pointer to notch count table. Bit 7 of the pointer is checked for zero; if no, the interrupt is not a marker interrupt so the functions performed by the ISS are same as for a typical slave phase already explained except that it even sends out notch count to cause marker interrupt. This can be seen in the flowchart Fig. 4.17.

If Bit 7 is set to zero, the program goes to marker. A test of change flag is done (details given in Sec. 4.6), if change flag is 00H the program jumps to NOCHANGE. In nochange the ISS accesses valid non-zero, appropriate notch counts in the lookup table within next 60 degree interval for all the phases and sends it to respective pulsewidth counters. The

term valid implies that for phase B,C a non-zero notch count preceeding a marker is not sent to notch counter as only master phase A can cause marker interrupts. The output pattern for all the phases is retrieved from table and set at appropriate bits for each phase at the output port. Then interrupt numbers for all phases are corrected, interrupts enabled before returning from the ISS.

If change flag is set to  $\emptyset 1H$ , then  $W_{tc}$  complement is sent to 8282 latches, new modulation index replaces old modulation index, change flag is reset before proceeding to nochange. The tasks accomplished at nochange is already explained.

If change flag is set to  $\emptyset 2H$ , implying change to SPWM mode of operation, a attempt flag which tells number of markers interrupts since change flag is recognised is tested. If zero, then all the activities of nochange is accomplished, interrupts enabled, sine pointers for synchronous SPWM are initialised with an additional 60 degree phase shift from the present status. The sine pointers are obtained from lookup table for interrupt number for phase A versus sine pointers for all three phases. The contents of this lookup table is given in Table 1 of this chapter. The read, write pointers are reinitialised and computations for pulsewidth counts for all the three phases required in carrier interval of synchronous

SPWM is completed. The attempt flag is set before returning to main program. It should be noted that change flag has as yet not been reset also normal operational of optimal PWM continues for an additional 60 degree interval i.e. till next marker interrupt is reached. At the next marker interrupt, the change flag being 02H and attempt flag being set the ISS goes to a change mode routine. This routine sends complemented  $W_{tc}$  to 8282 latches, replaces old modulation index by new modulation index, resets attempt and change flag. It then places addresses of ISS's for synchronous mode SPWM with ratio 18 of all the three phases at the respective interrupt address pointer before jumping to master phase ISS thus accomplishing the change. The flow chart for master phase ISS in optimal mode is given in Fig. 4.17.

#### 4.6 REQUIREMENTS AND IMPLEMENTATION OF MODE CHANGING:

Smooth changes between distinct modes of operation of the pulsewidth modulator is essential. Even for small voltage overshoots the current overshoots are very large Ref.[10,11].

In a closed loop system, the output voltage and frequency commands are obtained from feedback controllers. To avoid possibility of toggling between two adjacent modes due to transient oscillations in the feedback controller outputs, a hysteresis characteristics has been provided between



adjacent modes of the pulsewidth modulator. A hysteresis band of 1 Hz is considered suitable. Hence for a change from asynchronous SPWM to synchronous SPWM with ratio 36 the output frequency command should exceed 5.5 Hz. But for a change from synchronous mode SPWM with ratio 36 to asynchronous mode SPWM the output frequency command has to be less than 4.5 Hz. Such a hysteresis band has been provided between all adjacent modes as shown in Fig. 4.15.

The requirements for smooth and efficient transition between adjacent modes are enumerated below. But before that, how the controller recognises a change command, how it implements the hysteresis characteristics and what are the operations to be done before a change can be initiated (i.e. change flag set) is explained.

The new output frequency command is accepted by the main program Fig. 4.5. It then identifies the present mode of operation from a mode flag. Check with hysteresis characteristics whether the new output frequency command is acceptable i.e. not above higher mode upper hysteresis limit and not below the lower mode lower hysteresis limit. It checks whether mode barrier is crossed, if yes it sets the change and mode flags appropriately. Then depending on new mode of operation and frequency command  $F_s$ ,  $W_{tc}$  is calculated. If

new mode is asynchronous mode  $W_{tc}$  is constant and minimum but 'S' the sampling interval is determined from eqn. (4.9). If new mode is synchronous SPWM with ratio 36,  $W_{tc}$  is determined from eqn. (4.4). If new mode is synchronous SPWM with ratio 18, eqn. (4.7) is used to determine  $W_{tc}$ . Finally for optimal PWM,  $W_{tc}$  is determined from eqn. (4.10).

Due to the hardware implementation, the complement of  $W_{tc}$  is required, so the word  $W_{tc}$  is complemented and stored in scratch pad memory. The output voltage command is retrieved from V/F table and stored as new modulation index. Then depending on the type of mode change an appropriate mode flag is set. For changes within a mode, implying new mode is same as the old mode but  $F_s$  command is changed, the change flag is set to  $\emptyset 1H$ . For changes between asynchronous mode and synchronous mode SPWM also the change flag is set as  $\emptyset 1H$ . For mode change from synchronous mode with ratio 36 to synchronous mode with ratio 18, the change flag is set  $\emptyset 2H$ , but for the reverse it is set to  $\emptyset 1H$ . For changes between synchronous mode SPWM and optimal PWM the change flag is set to  $\emptyset 2H$ . The reason for keeping change flag at two different values is that for change flag =  $\emptyset 2H$  certain extra constraints and conditions have to be satisfied before a change of mode can be accomplished. These are elaborated below. During the operation of pulsewidth

modulator, the change flag is polled at appropriate interrupts i.e. sampling interrupt in asynchronous mode, carrier interrupt in synchronous mode and marker interrupts in optimal mode. If found set the constraints given below are ensured.

For change within asynchronous mode itself, new 'S' replaces the remainder and new modulation index replaces the old modulation index to effect change. The change from asynchronous mode to synchronous mode SPWM can be accomplished at the end of a sampling interval. The complemented  $W_{tc}$  word calculated for new output frequency is sent to 8282 latches, new modulation index replaces old modulation index and interrupt address pointer is directed to new mode ISS.

For change within synchronous mode SPWM with ratio 36, complemented  $W_{tc}$  is sent to 8282 latches, new modulation index replaces old modulation index. For change to asynchronous mode complemented  $W_{tc}$  (this is constant for asynchronous SPWM) is sent to 8282 latches and new sampling interval 'S' replaces remainder. The interrupt address pointer for master phase A is pointed to carrier ISS for asynchronous SPWM. For a change from synchronous SPWM with ratio 36 to synchronous SPWM with ratio 18, the sine pointers for all phases are made even for ease of implementation, complemented  $W_{tc}$  is sent to 8282 latches, new modulation index replaces old modulation index.

For a change within synchronous mode with ratio 18, or mode change to synchronous mode with ratio 36, complemented  $W_{tc}$  is sent to 8282 latches, new modulation index replaced old modulation index, new ISS address placed at interrupt address pointer.

Mode change between synchronous SPWM and optimal PWM is more involved. This is basically due to the totally different philosophies of the modes. The optimal PWM scheme implemented in this thesis is self starting scheme at 'markers' in the optimal waveform. These markers are at 60 degree points of the waveform. Hence mode changes can occur at 60 degree points of output voltage waveform.

Change of mode from synchronous SPWM to optimal PWM can occur at any 60 degree point of sine lookup table which is used to generate the PWM trigger signal. Upon recognition of change, depending on the status of the output waveform currently being generated, represented by phase sine pointers, the interrupt numbers for the three phases are initialised. The complemented  $W_{tc}$  is sent to 8282 latches, new modulation index replaces old modulation index. The interrupt address pointers for all the phases are pointed to optimal PWM ISS for respective phases. The program then jumps to master phase 'marker' interrupt ISS thus accomplishing change. The lookup

table for sine pointer phase A to interrupt numbers for optimal mode is given in Table 1.

A change within optimal PWM is effected by sending, new complemented  $W_{tc}$  to 8282 latches and new modulation index replacing old modulation index.

A mode change from optimal PWM to synchronous SPWM is quite complex. The optimal PWM is a lookup table, pattern retrieval technique and synchronous SPWM is a carrier by carrier on-line computational scheme. The computations for synchronous SPWM are carried out one carrier cycle in advance. When a change from optimal mode to synchronous mode is desired, to avoid transient voltages, the delays for the correct sine sample should already be computed and available to be output to pulsewidth counters. Hence depending on the present status of optimal output waveform being generated, represented by phase interrupt numbers, the sine pointers for the three phases are delayed by 60 degrees to the present status and computations for pulsewidth counts initiated. In the meanwhile the optimal PWM mode is allowed to continue unhindered for another 60 degree. The structure of the pulsewidth modulator being interrupt based both these task are accomplished together. Upon reaching the next marker interrupt, the interrupt service

subroutines for synchronous mode with ratio 18 are enabled by placing appropriate addresses at interrupt address pointer of all the phases. The program then jumps to master phase carrier ISS to accomplish the mode change.

The above explained aspects have been shown in flow-charts for the master phase A for different modes as already explained in Section 4.3 to 4.5.

#### 4.7 LOOKUP TABLES REQUIRED BY THREE PHASE PULSEWIDTH MODULATOR:

The optimal PWM mode of operation is a lookup table and pattern retrieval technique. It requires the notch count table and the output trigger pulse pattern table. A sine lookup table is required for computations of pulse counts in all modes of SPWM. For the open loop operation of pulsewidth modulator a voltage-frequency lookup table is stored. Lookup tables are required to efficiently accomplish mode change between synchronous SPWM and optimal PWM.

The major tables are elaborated upon here. The mode changing lookup table and lookup table for optimal mode PWM are given in Table 2.

##### 4.7.1 Notch Count Lookup Tables for Optimal PWM:

The process of optimisation of the optimal PWM pattern is time consuming. Hence the switching angles for only a

limited critical output voltage values are determined. Interpolation is carried out between these computed points in desired step size of output voltage to determine switching angles. These angles are linearly translated into notch counts. For the lookup table in this thesis, a 60 degree interval is represented as 256 (0FFH) clock states. For the optimal waveform of 3 notches/quarter cycle selected, Fig. 4.14, notch counts for the tables are determined as  $W_1 \alpha \alpha_1$  degrees,  $W_2 \alpha (60 - \alpha_1)$  deg.,  $W_3 \alpha (\alpha_2 - 60)$  deg.,  $W_4 \alpha (\alpha_3 - \alpha_2)$  deg. and  $W_5 \alpha (90 - \alpha_3)$  degrees. With linear scaling the notch counts are stored in lookup table.

The notch count table accessed during the operation of the pulsewidth modulator depends on the status of output voltage waveform. The notch count accessed within the table is determined by desired output voltage.

In implemented scheme, optimal PWM is implemented beyond 19.5 Hz. A voltage-frequency table has been stored for open loop operation. Hence optimal PWM is implemented for output voltage 44.14% to 100% in steps of 0.39%. Hence a total of 143 notch counts are stored in each notch count table.

#### 4.7.2 Voltage-Frequency Lookup Table:

In a closed loop control of induction motor, independent control of voltage and frequency is desired. But for open

loop control, the voltage and frequency are linearly related except at very low frequencies where boosting of voltage is required and for frequencies above rated where the rated voltage has to be held constant. This characteristics has been stored in the V/F lookup table.

In the implemented scheme, desired output frequency is the command to the system. This is used as a pointer to access the desired output voltage from V/F lookup table. This is the new output voltage or new modulation index command. The size of V/F table is 256 locations, the maximum frequency command is  $F_S = 2048$  (decimal) hence software manipulation is required to retrieve the correct output voltage command.

#### 4.7.3 Sine wave Lookup Table:

The sine samples required for computations of pulse-width counts in regularly sampled SPWM are accessed from the sine lookup table. The number of sine samples required is equal to the maximum ratio of carrier frequency to modulating frequency ratio ( $n$ ).

In the implemented scheme  $n=36$ , hence 36 samples of an inverted sine wave of unit amplitude are stored in the lookup table. A scaling factor is used in which 1 corresponds to 127 decimal (7FH). The sign magnitude binary representation is used to represent the sine samples as it suits best for the



multiplication operation required for computation of pulsewidth count. The inverted sine samples (sine samples stored in reverse order) are stored with least significant address of first location 01H for ease of software manipulation. The sine pointers for the three phase are initialised as sine pointer phase A = 24H, sine pointer phase B = 18H, sine pointer phase C = 0CH. With this the correct phase relationship is initialised for the three phases.

#### 4.8 DISCUSSION AND EXPERIMENTAL RESULTS:

The number of interrupts have been reduced using master phase concept. This helps to avoid the possibility of deliberate pending of interrupts for any phase. It also helps in smooth and efficient mode transitions.

The execution time delay of program before a pulsewidth counter is reloaded as well as setting output trigger signal at output port affects the accuracy of the PWM output frequency. It becomes prominent as the ratio  $n = f_c/f_m$  is increased. This problem of execution time delay can be overcome by having hardwired logic gating for deriving output pulses. The pulsewidth counter can then be loaded with pulsewidth or notch count in advance. Same effect as obtained by computation of the pulsewidth counts one carrier in advance. Even as the present count is converted into delay, the next count is sent to the counter input/output buffer. The counter is configured

programmed to give an output pulse upon reaching terminal count and then automatically loading the count from the I/O buffer and proceeding unhindered (Mode 2 operation in Intel 8253 counters). The output pulse is used as an interrupting pulse, hence all, interrupts have to be edge sensitive.

For better performance in asynchronous mode, the hard-wired technique for sampling interval should be used. Infact the same sampling interval counters clocked by high frequency clock source can be used to generate the carrier interrupt in the synchronous mode. With this a fine resolution in output frequency can be achieved.

All these schemes are hardware solutions which usually are faster. But the improved performance is obtained at the price of increased hardware burdens.

The experimental results are shown in the form of photographs of the gating trigger pattern from the three phase pulsewidth modulator. The explanation of waveforms in the photograph are given below each photograph. These are from Fig. 4.19 onwards.

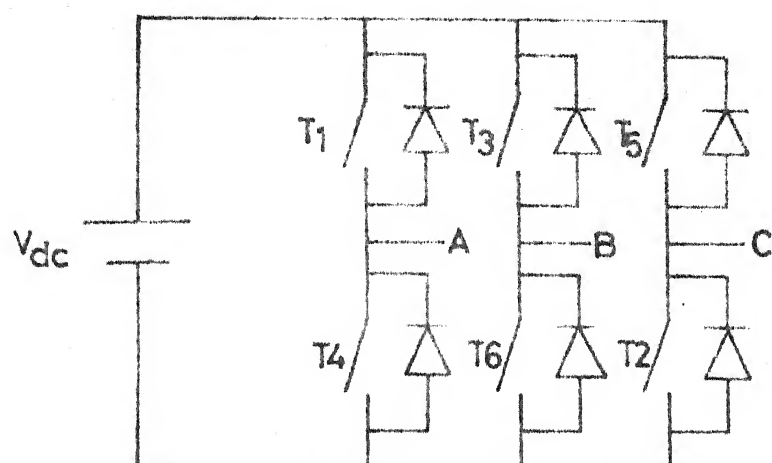


Fig 4.1a Basic 3 phase PWM Inverter Configuration

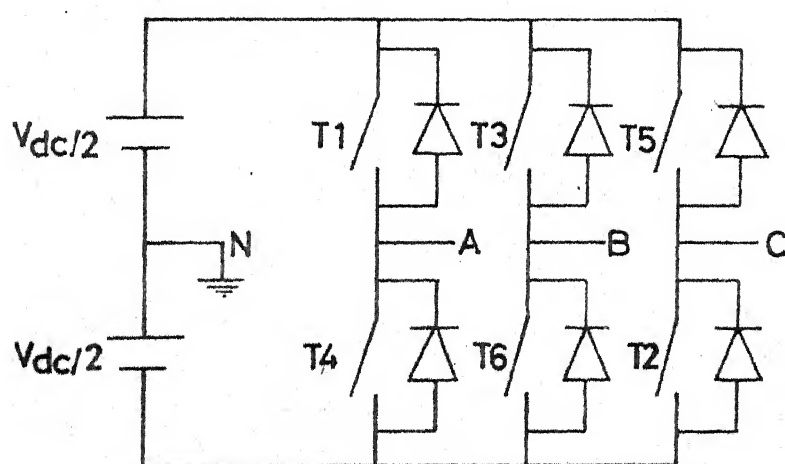


Fig 4.1b Schematic representation with Split Supply

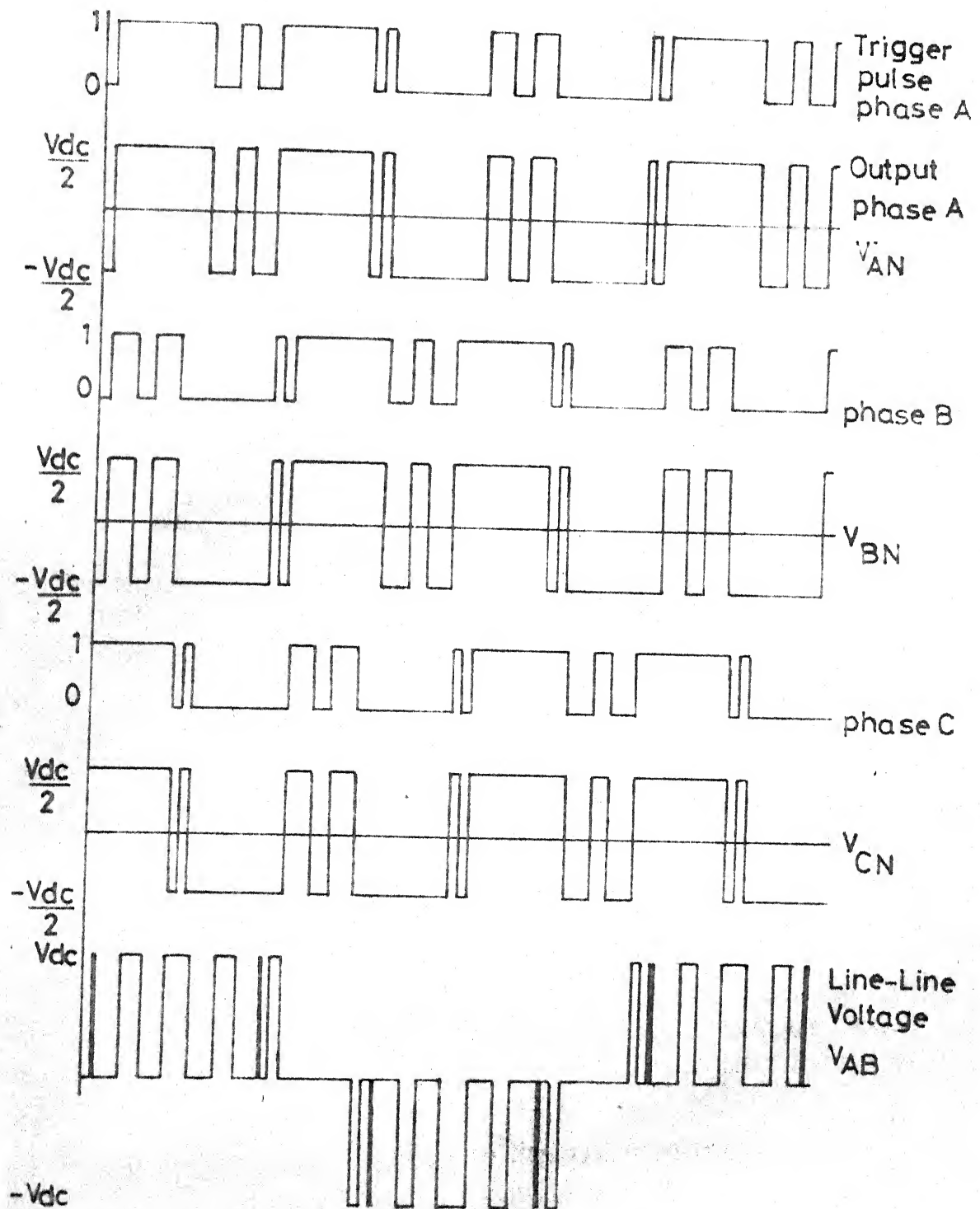


Fig 4.2 Trigger pulse, phase voltage, line to line voltage

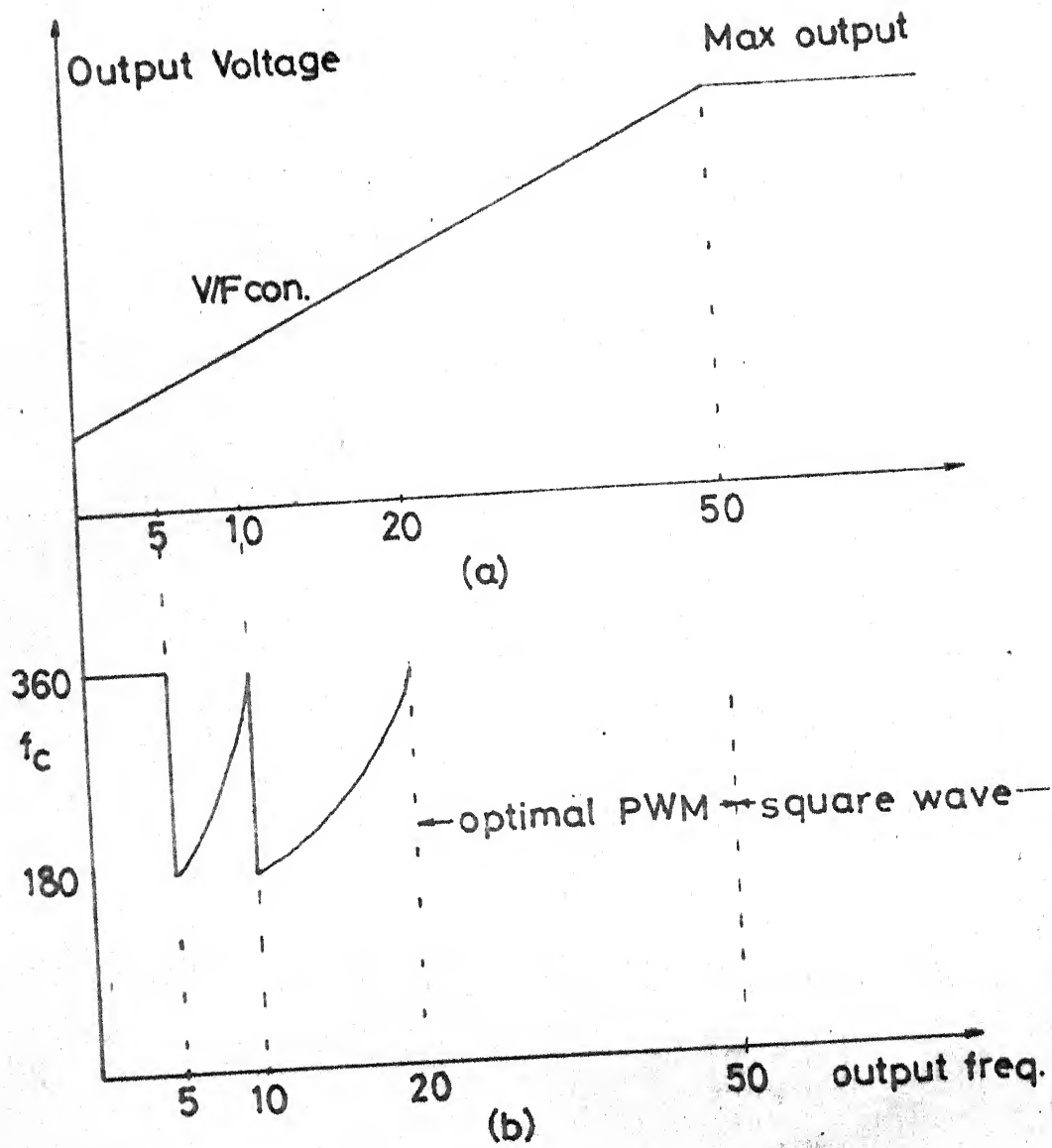


Fig 4.3 (a) Output Voltage (b) Carrier frequency versus Output frequency

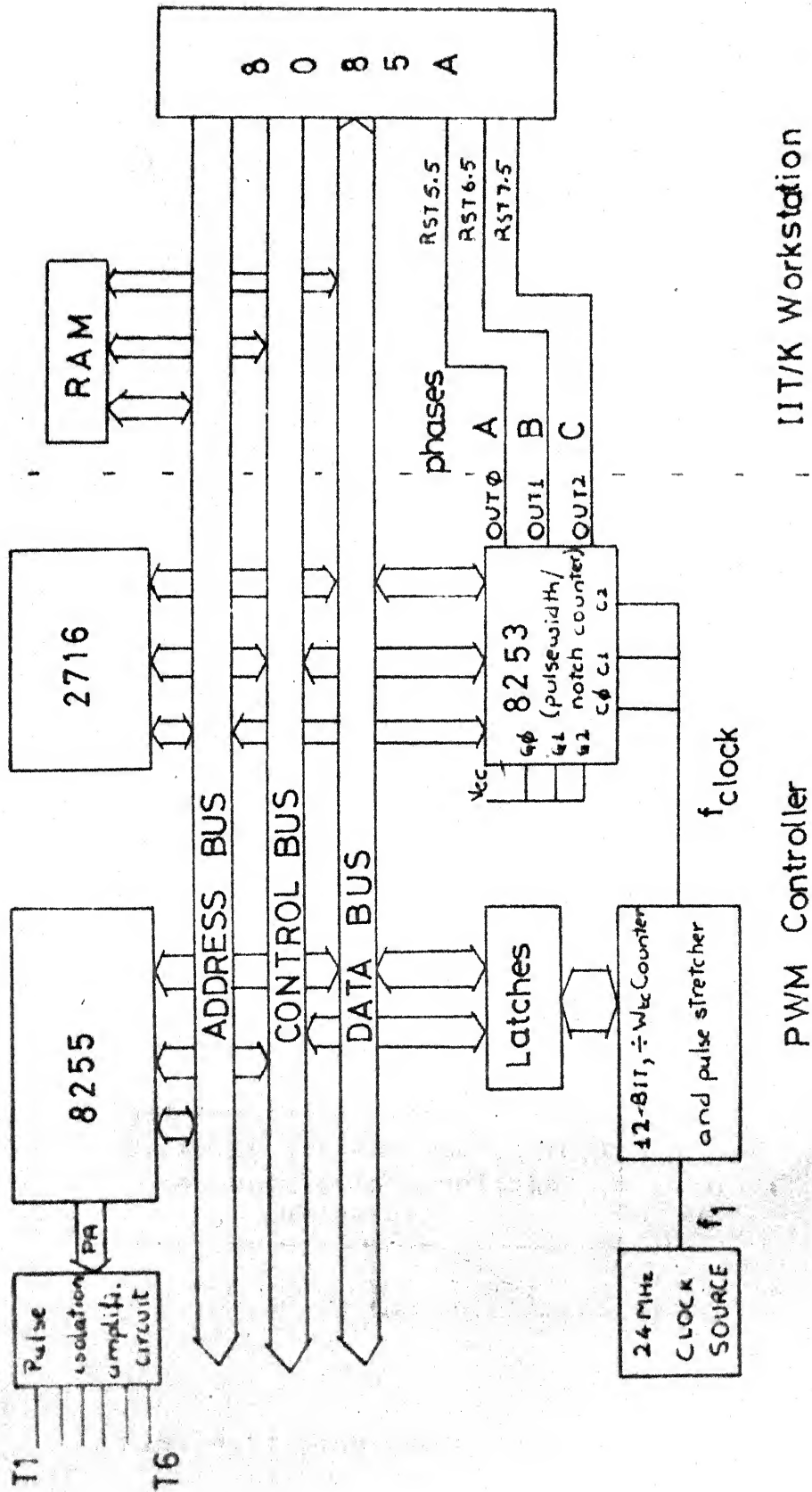


Fig 4.4 Block diagram of 3-Phase Pulsewidth Modulator

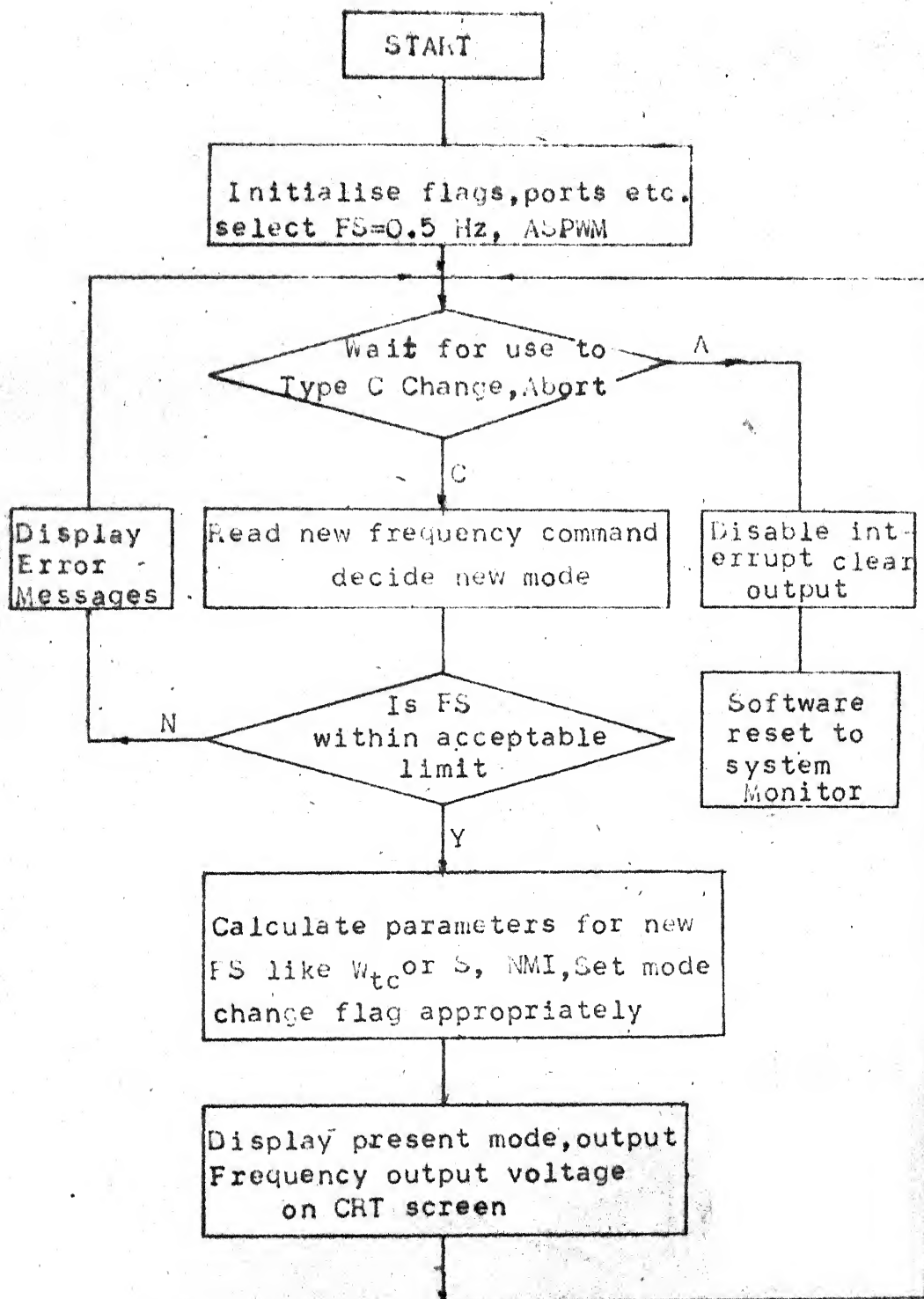


Fig. 4.5: Background Routine

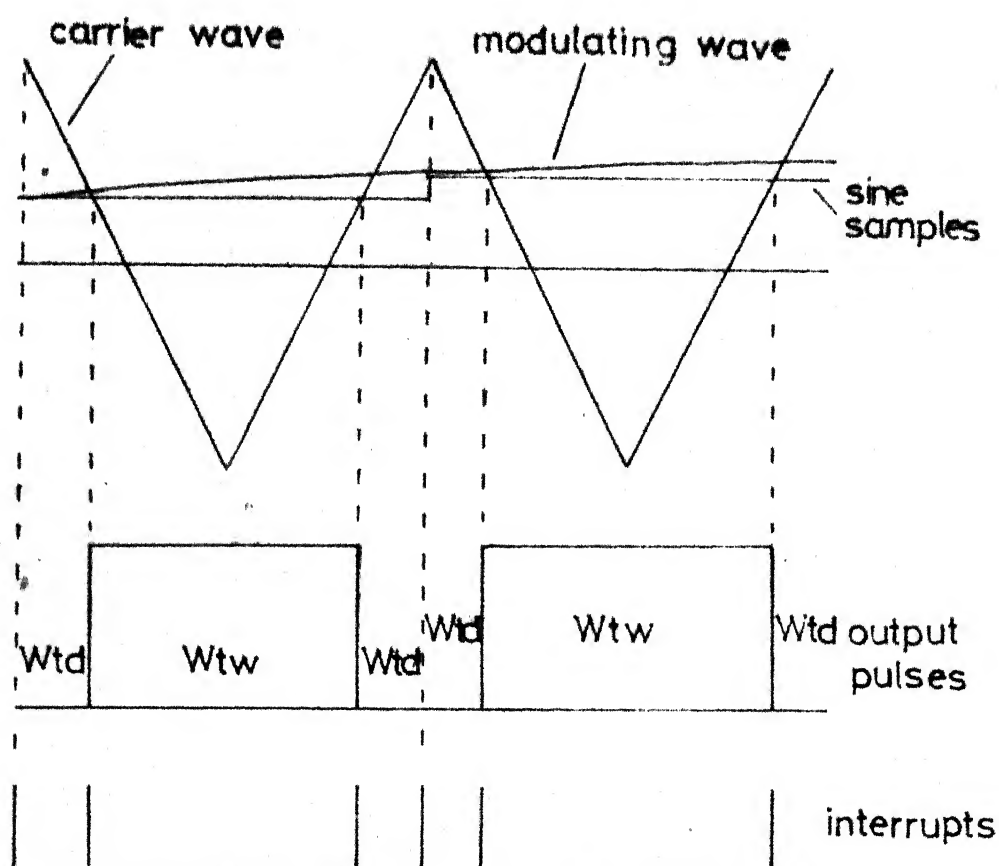


Fig 4.6 Regular Symmetric Sampled PWM Trigger Pulse

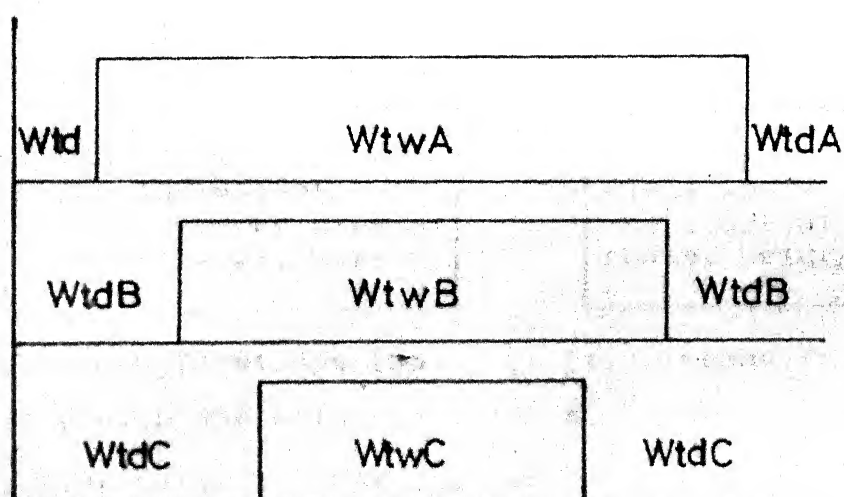


Fig 4.7 3-Phase PWM Trigger Pulses in a Carrier interval



Correct read pointer  
IADPTA and access  
 $W_{tw}$  delay for phase

Output delay to  
phase pulsewidth  
counter

Set output pins for phase  
 $\phi 1$ . Place address of  
second ISS present mode at  
IADPT. Return

(1) First Interrupt ISS

Correct head pointer  
access and output  
 $W_{tw}$  delay for phase

Send output for  
phase  $\phi 1$  bits

Place address of second  
interrupt at IADPT for  
phase, Return

(1) First Interrupt ISS

Correct head pointer  
IADPTA, access  $W_{tdA}$   
delay, output to counter

Set output pins for  
phase A  $\phi 1$

Place address of  
carrier ISS of present  
mode at IADPT. Return

(ii) Second Interrupt ISS

a) Master phase A ISS

Correct read pointer  
access and output  
 $W_{td}$  delay for phase

Send output for  
phase  $\phi 1$  bits

Place address of first  
interrupt at IADPT for  
phase. Return

(ii) Second Interrupt ISS

b) Typical slave phase ISS

Fig. 4.8: Typical First and Second Interrupt ISS for SPWM

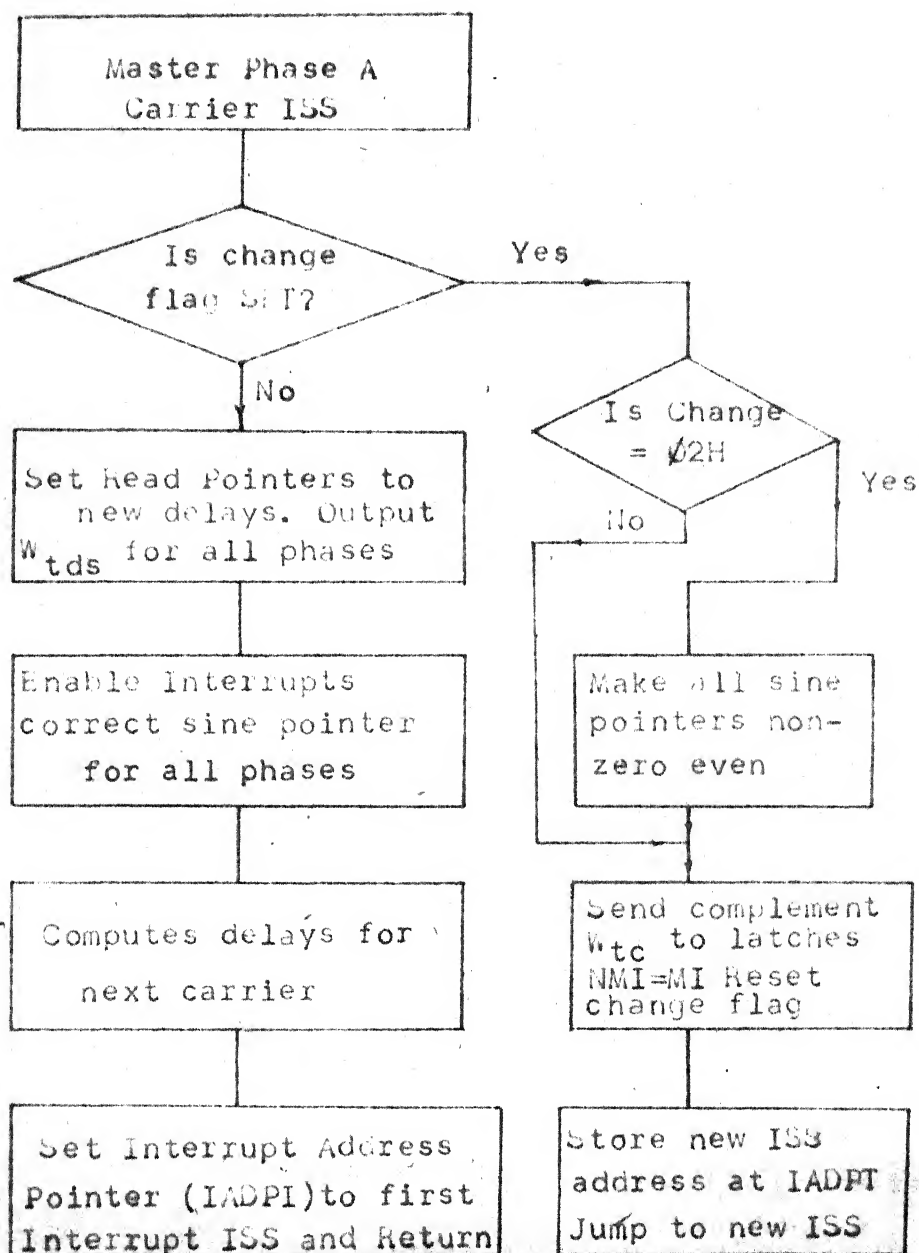


Fig. 4.9: Carrier ISS for Sync. mode ratio 36.

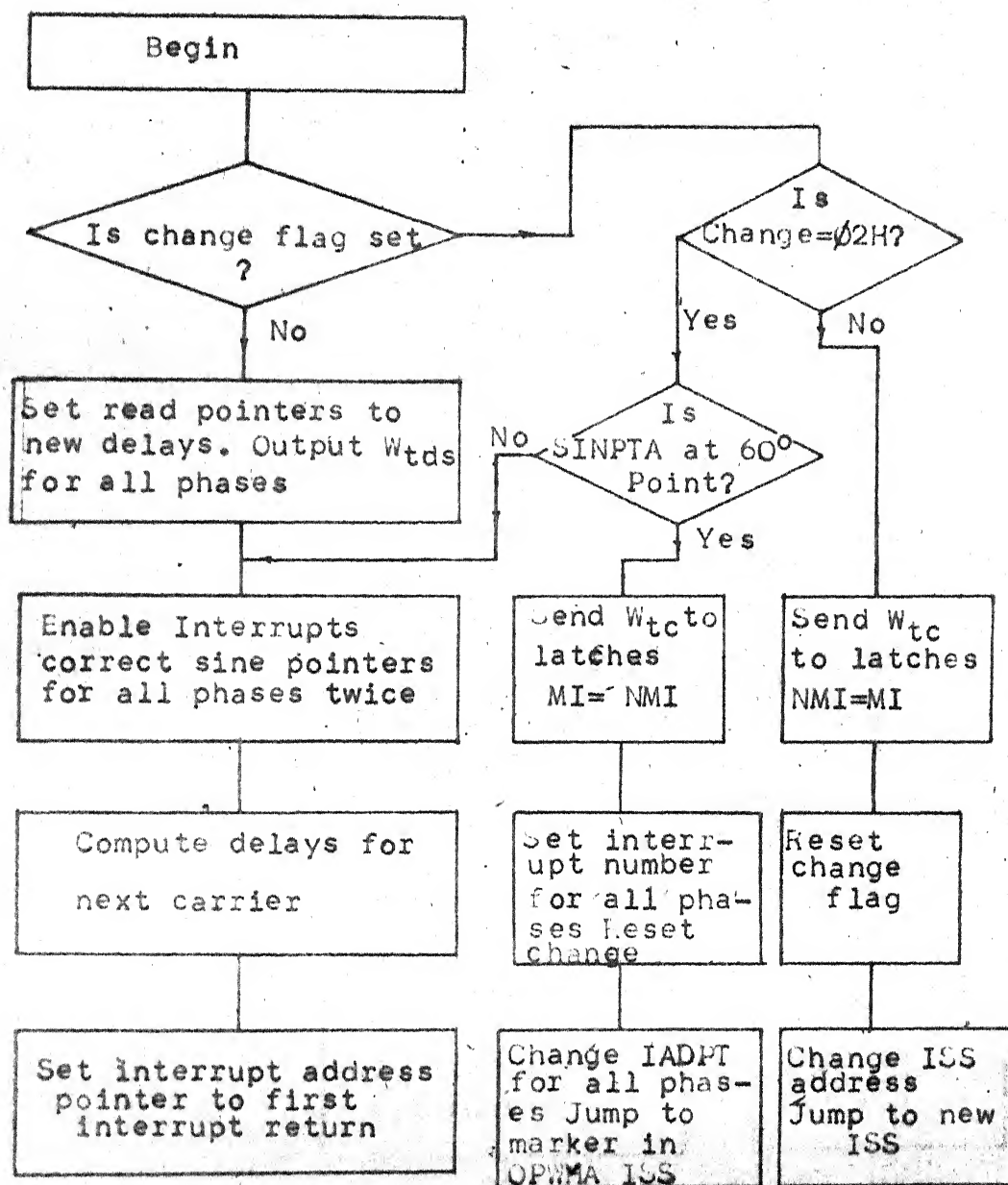


Fig. 4.10: Carrier ISS for Sync. mode ratio 18

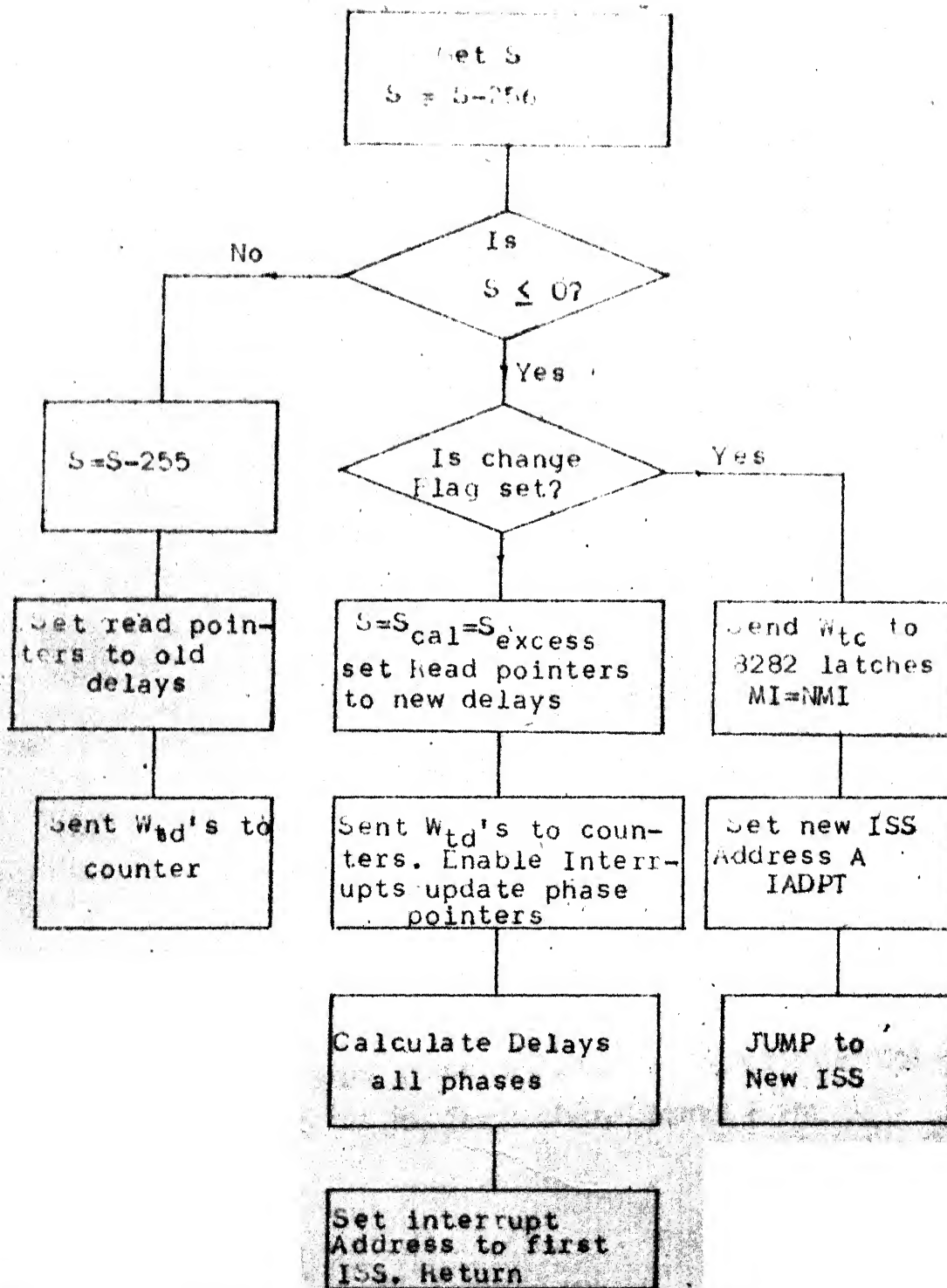


Fig. 4.11: Carrier ISS Async. Mode

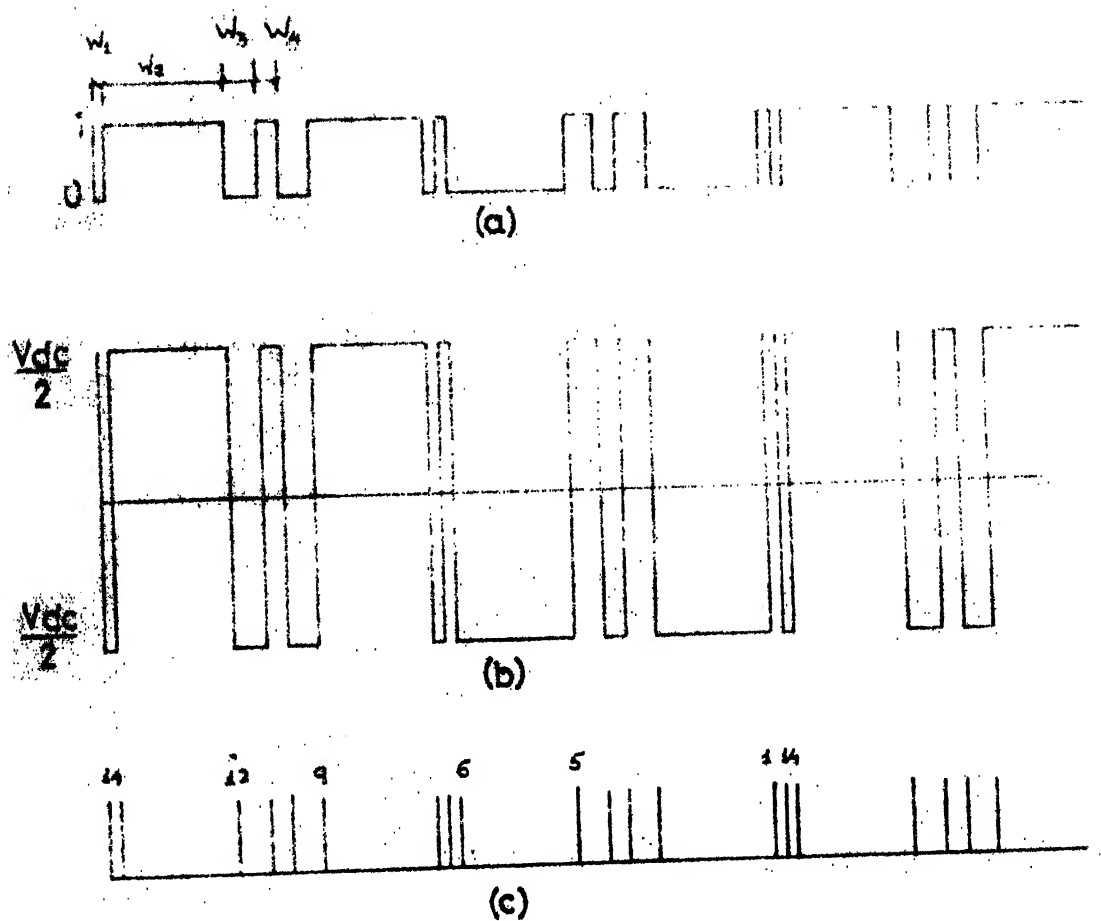
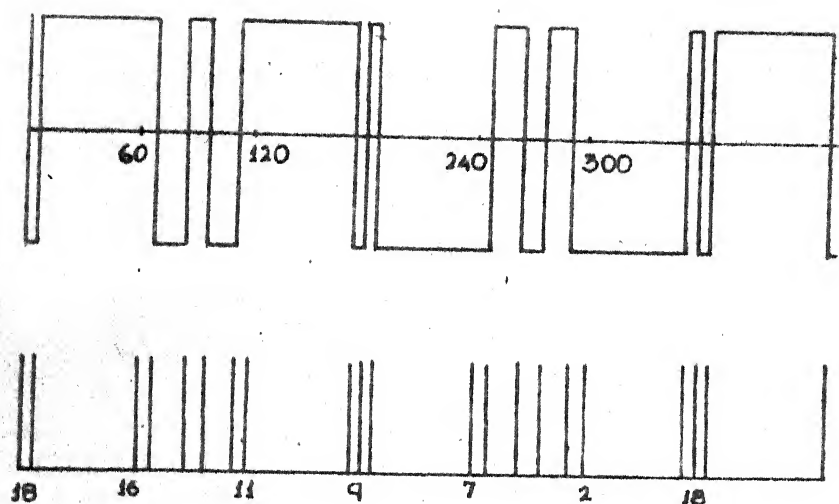
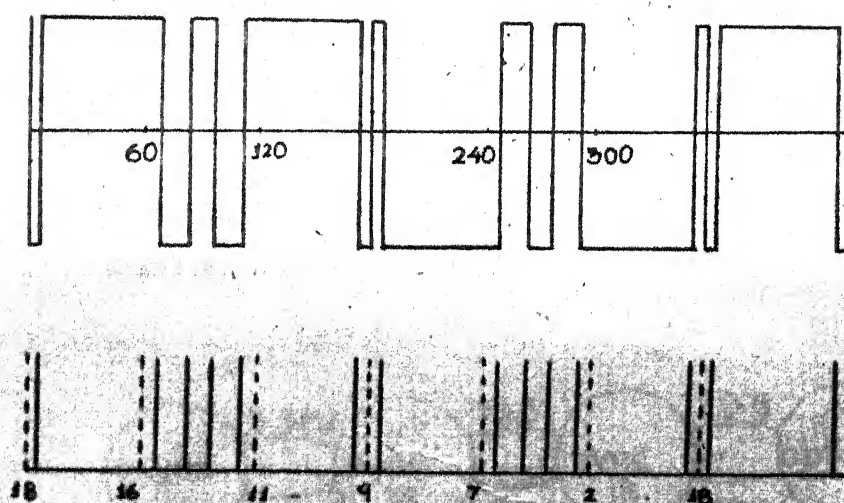


Fig 4.12 (a) Trigger pulse, (b) Output voltage, (c) Interrupt levels for Single phase Optimal PWM



a. Master phase output with interrupts



b. Slave phase output with interrupts  
(dotted marker interrupts not caused)

Fig 4.13 Optimal PWM Implementation

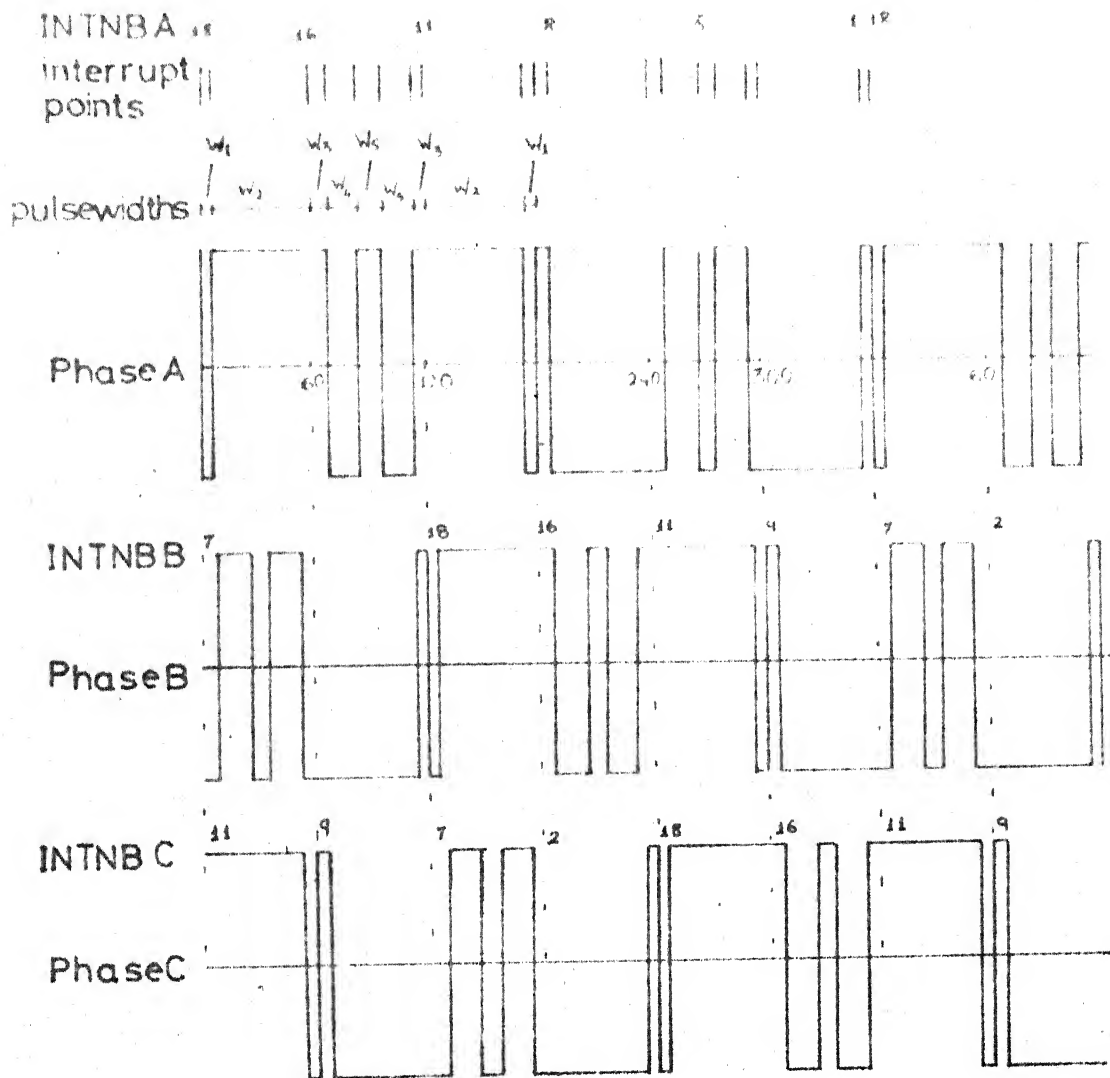


Fig 4.14 3 Phase Optimal PWM with delay widths and interrupt numbers

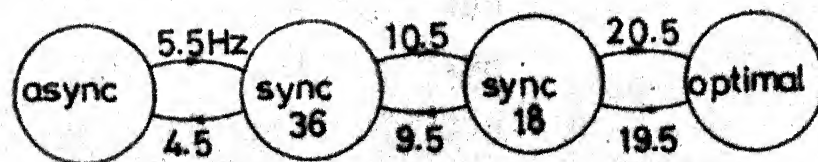


Fig 4.15 Mode Change Flow Diagram

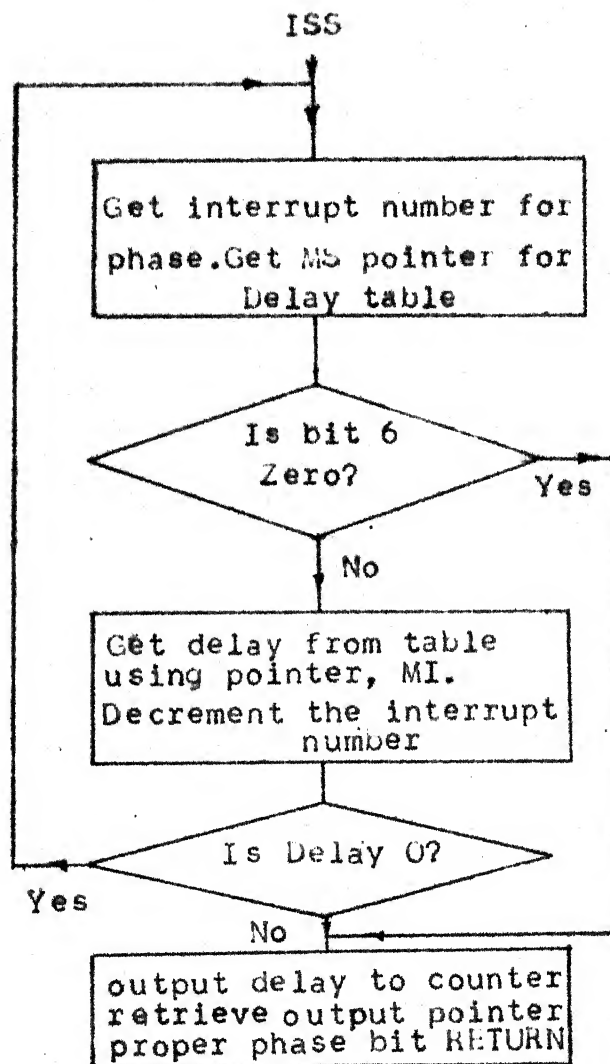


Fig. 4.16: ISS for typical slave phase in optimal PWM

Table 1: Lookup table for sync. mode to optimal transition

SINPTA	INTNB A	INTNB B	INTNB C
24H	12H	07H	0BH
1EH	10H	02H	09H
18H	0BH	12H	07H
12H	09H	10H	02H
05H	07H	0BH	12H
00H	02H	09H	10H

INTNB= INTERRUPT  
NUMBER



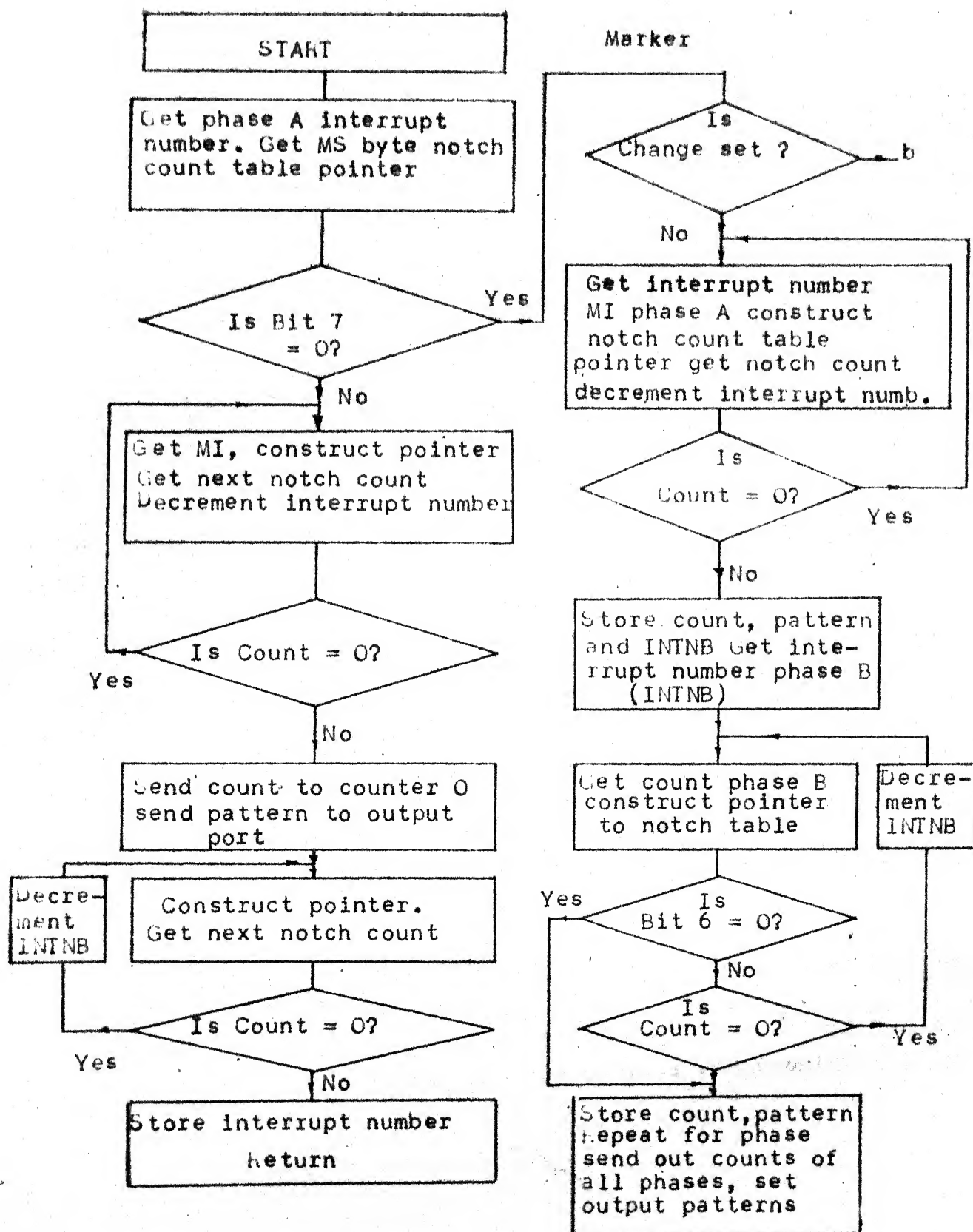


Fig. 4.17(a): Master phase 155 for optimal PWM

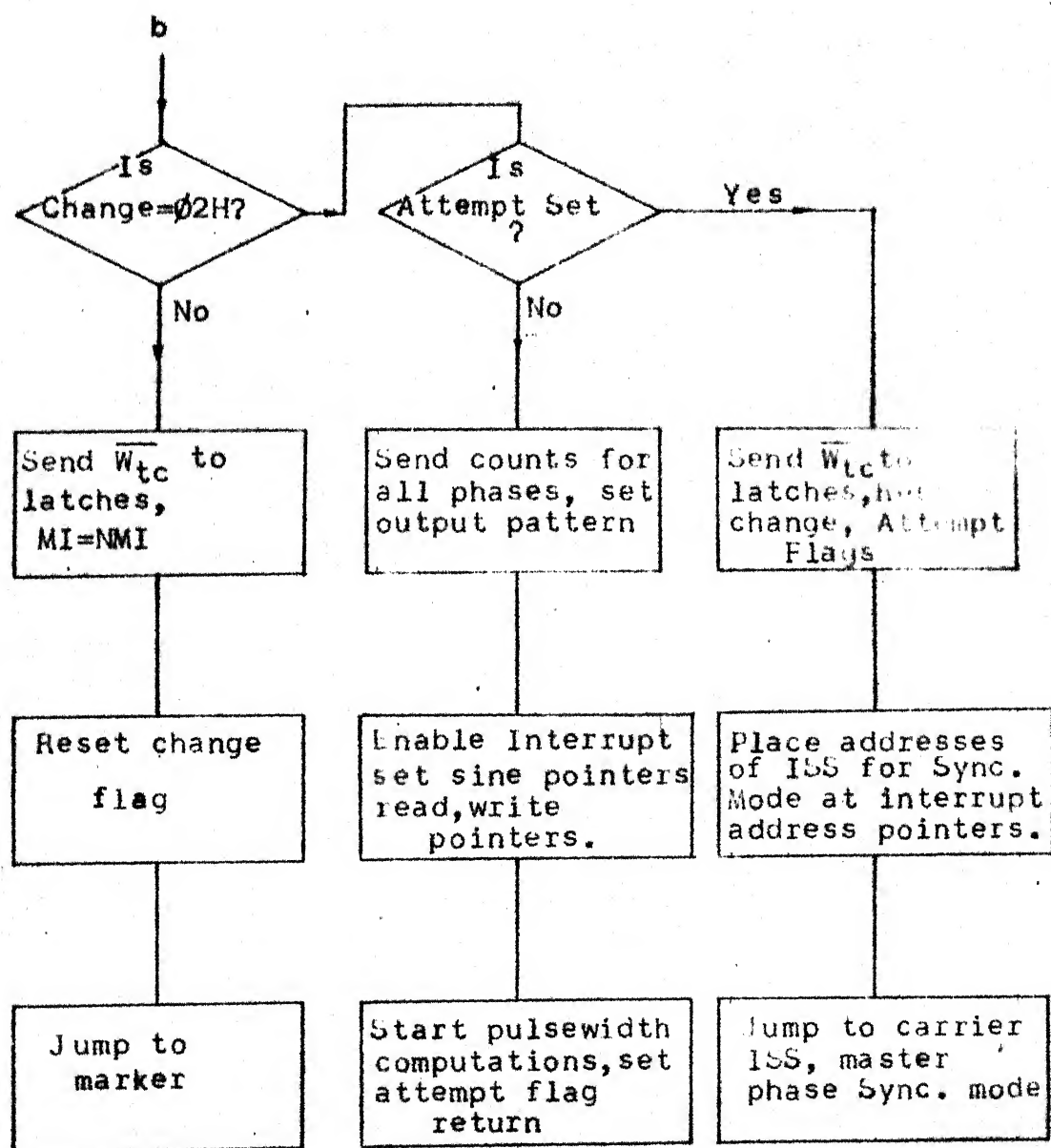


Fig. 4.17(b) Master Phase ISS for Optimal PWM (contd..)

For the scheme implemented

$$f_{S \max} = 100 \text{ Hz represented by } F_{S \max} = 2048 = \text{07FF (Hex)}$$

$$f_1 = 24 \text{ MHz, } N = 256 = \text{0FF (Hex), } n = 36$$

i) For synchronous SPWM with ratio 36 for  $f_S = 7.5 \text{ Hz}$

$$W_{tc} = \frac{F_{S \max}}{f_{S \max}} \cdot \frac{f_1}{n \cdot F_S} \cdot \frac{1}{N} = \frac{2048}{100} \cdot \frac{24 \cdot 10^6}{36 \cdot 154} \cdot \frac{1}{256}$$

$$= 346 = \text{015A (Hex)}$$

ii) For synchronous SPWM with ratio 18 for  $f_S = 15 \text{ Hz}$

$$W_{tc} = \frac{F_{S \max}}{f_{S \max}} \cdot \frac{f_1}{n' \cdot F_S} \cdot \frac{1}{N} = \frac{2048}{100} \cdot \frac{24 \cdot 10^6}{36 \cdot 154} \cdot \frac{1}{256}$$

$$= 347 = \text{015B (Hex)}$$

iii) For optimal PWM for  $f_S = 25 \text{ Hz}$

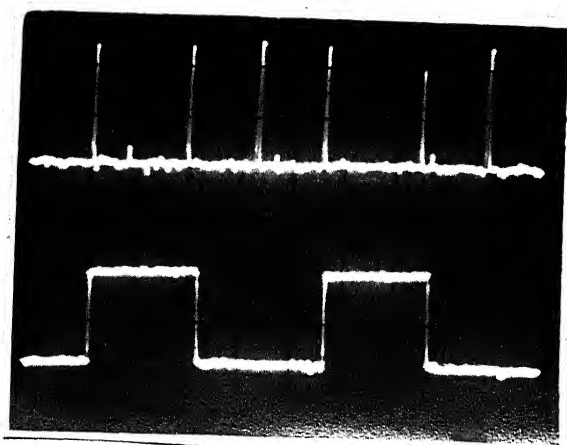
$$W_{tc} = \frac{F_{S \max}}{f_{S \max}} \cdot \frac{f_1}{6 \cdot F_S} \cdot \frac{1}{N} = \frac{2048}{100} \cdot \frac{24 \cdot 10^6}{6 \cdot 512} \cdot \frac{1}{256}$$

$$= 625 = \text{0271 (Hex)}$$

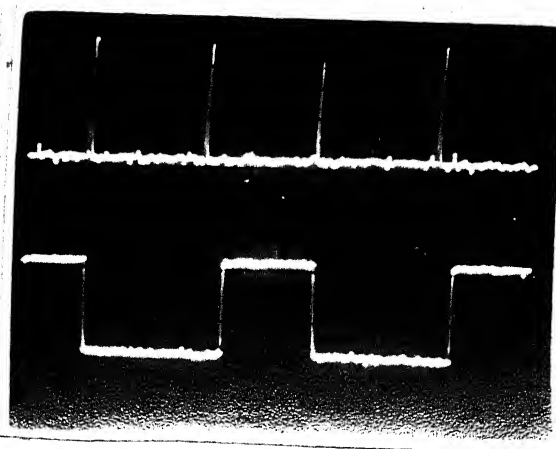
Fig. 4.18: Illustration of  $W_{tc}$  calculation.

Table 2: Table for INTNB v/s notch count pointer, output pattern, SINPT'S for mode changing.

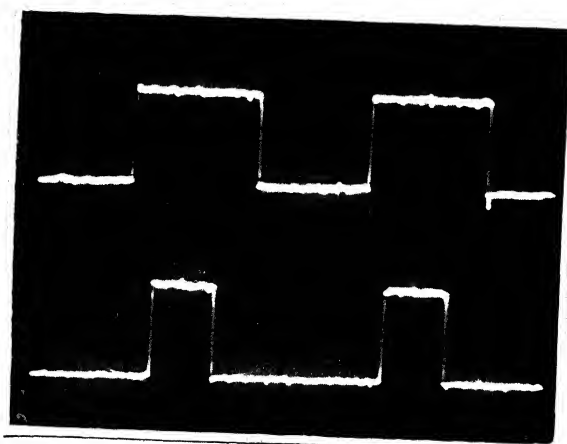
INTNB	12H	11H	10H	0FH	0EH	0DH	0CH	0BH	0AH	09H	08H	07H	06H	05H	04H	03H	02H	01H
Most significant byte delay pointer	75H	E6H	77H	F8H	F9H	F8H	B7H	76H	B5H	75H	E6H	77H	F8H	F9H	F8H	B7H	76H	B5H
Output for delay interval	0	1	1	0	1	0	1	1	0	1	0	0	1	0	1	0	0	1
SINPTA	1EH		18H				12H			0CH		06H					24H	
SINPTB	12H		0CH				06H			24H		1EH					18H	
SINPTC	06H		24H				1EH			18H		12H					0CH	



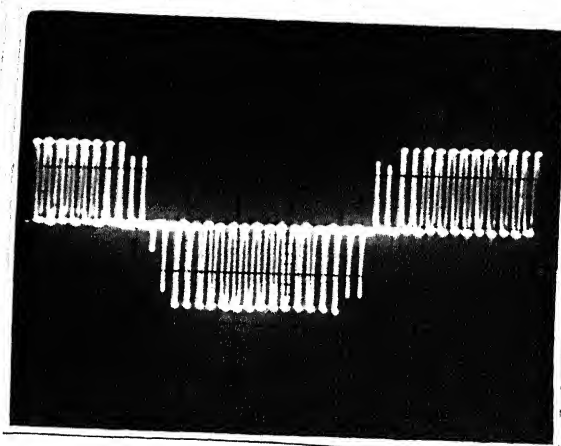
a) Master Phase A output trigger pulses with interrupt signals  
(time base: 1 msec/cm)



b) Slave Phase B output trigger pulses with interrupt signals  
(time base: 1 msec/cm)

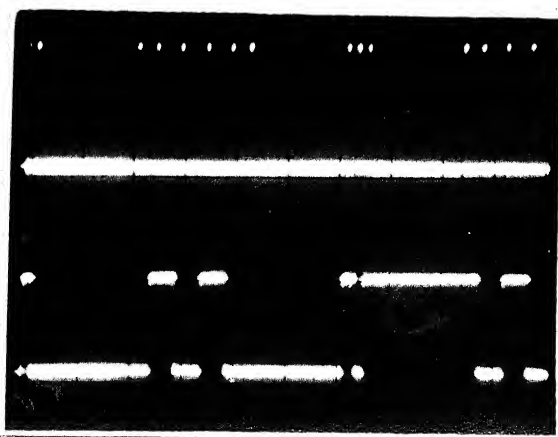


c) Output trigger pulses of two phases  
(time base: 1 msec/cm)

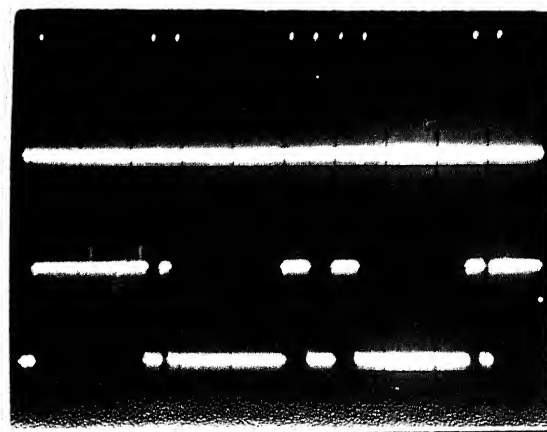


d) Line to line voltage synchronous mode ratio 18  
(time base 5 msec/cm)

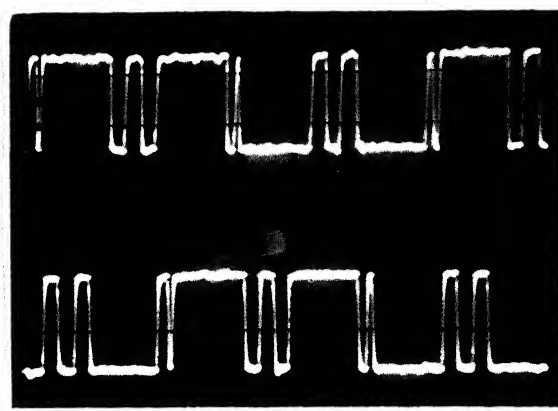
Fig. 4.19: Photographs of SPWM Implementation



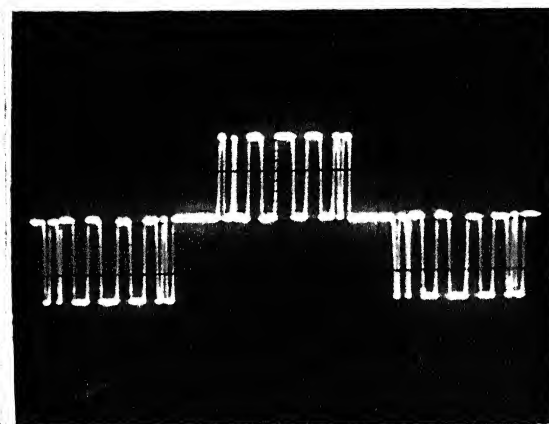
a) Master Phase A output  
trigger pulses with  
interrupt signals  
(time base 2 msec/cm)



b) Slave phase C output  
trigger pulses with  
interrupt signals  
(time base 2 msec/cm)

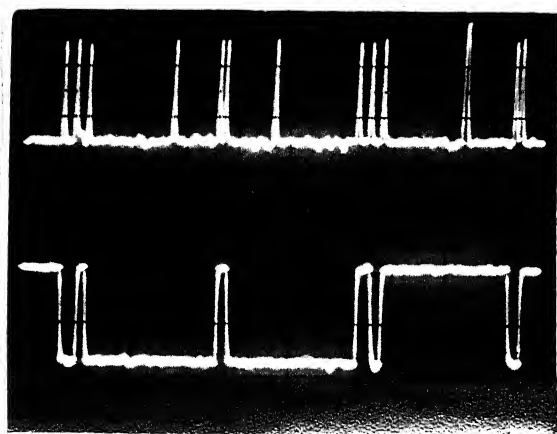


c) Output trigger pulses  
of two phases  
(time base 5 msec/cm)

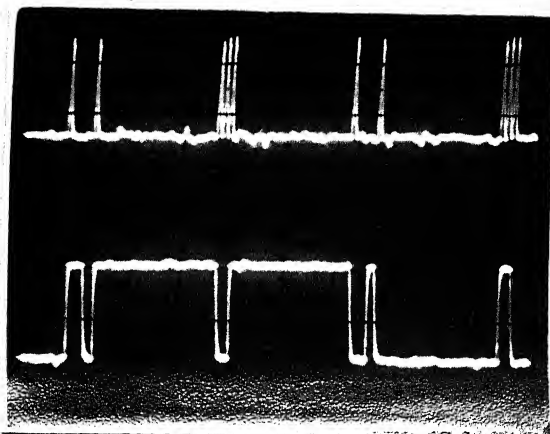


d) Line to line voltage  
waveform  
(time base 5 msec/cm)

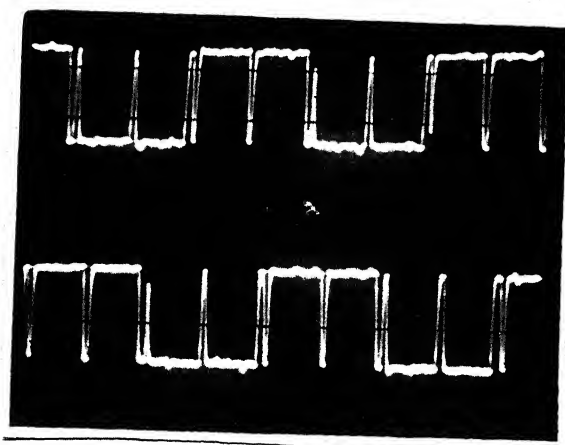
Fig. 4.20: Photographs of Optimal PWM Implementation



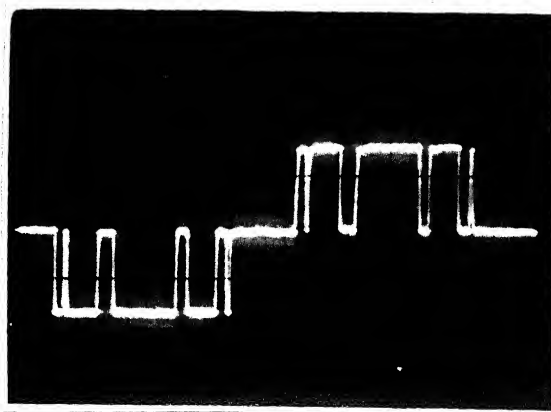
a) Master Phase A output  
trigger pulses with  
interrupt signals  
(time base 2 msec/cm)



b) Slave Phase B output  
trigger pulses with  
interrupt signals  
(time base 2 msec/cm)

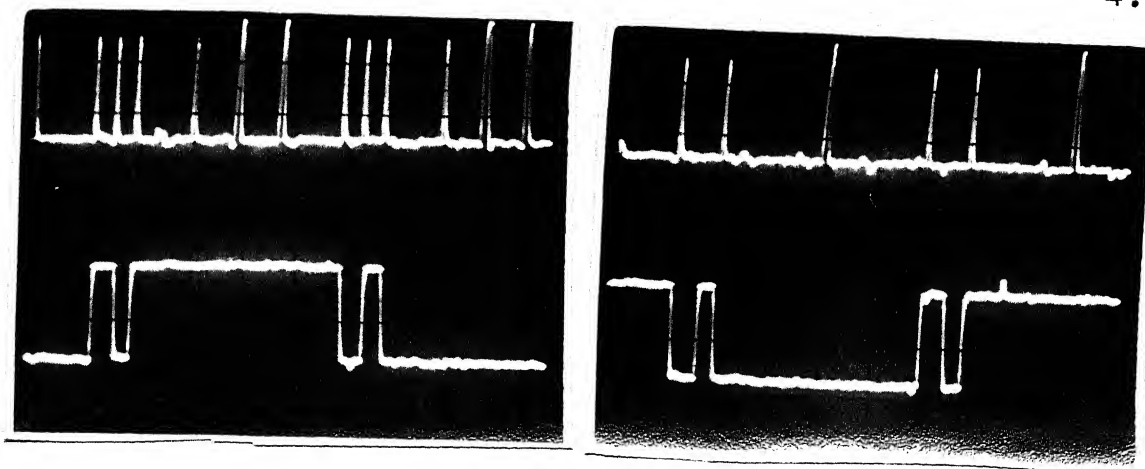


c) Output trigger pulses  
of two phases  
(time base 5 msec/cm)



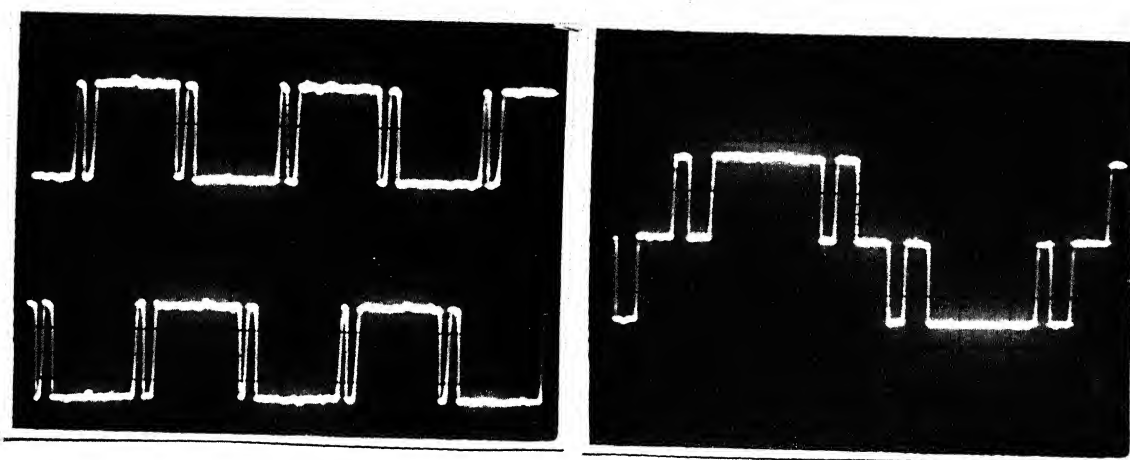
b) Line to line voltage  
waveform  
(time base 2 msec/cm)

Fig. 4.21: Photographs of Optimal PWM with one notch dropped.



a) Master Phase A output  
trigger pulse with  
interrupt signals  
(time base 1 msec/cm)

b) Slave phase C output  
trigger pulse with  
interrupt signals  
(time base 1 msec/cm)

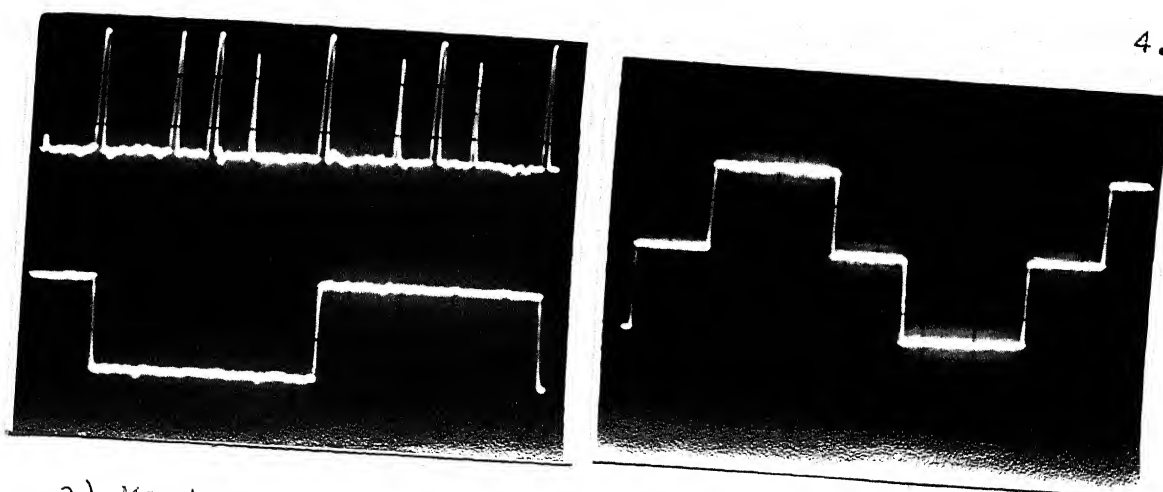


c) Output trigger pulses  
of two phases  
(time base 5msec/cm)

d) Line to line voltage  
waveform  
(time base 2 msec/cm)

Fig. 4.22: Photographs of optimal PWM with two notches dropped.





a) Master phase A output  
trigger pulses with  
interrupt signals  
(time base 2 msec/cm)

b) Line to line voltage  
wave form  
(time base 2 msec/cm)

Fig. 4.22: Photographs for square wave mode

## CHAPTER 5

### DESCRIPTION OF EXPERIMENTAL HARDWARE

In this chapter, a brief description of the host microprocessor (i.e. IITK workstation) hardware as well as the software is presented. The circuit developed to implement the controller for three phase bridge converter (Chapter 2) and the pulsewidth modulator (Chapter 4), is described. The details of circuit diagram and printed circuit layouts are given in the Appendix A. A very brief description of the auxiliary hardware developed is also given.

#### 5.1 DESCRIPTION OF HOST MICROPROCESSOR WORKSTATION:

The microprocessor workstation used in this thesis has been developed by the Electrical Engineering Department, Indian Institute of Technology, Kanpur. A brief description of the hardware and software provisions is given below.

The workstation has a ASCII keyboard and CRT display for human interface. The workstation has three printed circuit cards namely monitor card, I/O card and processor card. The monitor card has its own CPU and performs the functions of keyboard scanning and CRT refresh. This gives real time operation capability to the microprocessor on the processor card. It is interfaced with the CPU on the processor card.

via a RS232C serial link. The I/O card has provisions for analog to digital conversion and digital to analog conversion. It has few parallel ports, RS232C serial link and 8253 counters for users. The provisions on the I/O card are not used with the hardware developed in this thesis. Instead a separate controller card has been developed to take its place.

The processor card has 6K bytes RAM for user and has a powerful monitor program. Nearly all the lines of the processor 8085A are buffered and provided on an edge connector. The signals at the edge connector is given in Fig. 5.1.

The monitor program has facilities for assembly/disassembly of program, text, editor, system routines like READ, PRINT, BIHEX which accomplish communication with the monitor CPU through RS232C link, resulting in communication with user via keyboard and CRT display.

For a hardware to be used in conjunction with the workstation, the signals used by the printed circuit board developed should be compatible with edge connector signals provided by the workstation as already given (Fig. 5.1). The printed circuit board is inserted into the edge connector of the workstation. The microprocessor on the processor card is used to run the user programs. If the user programs are stored in the ROM, the ROM has to be placed on the user printed

circuit board. However the workstation RAM can always be used as scratch pad memory. The powerful monitor program of the workstation helps in easy debugging of software and the hardware developed by user.

In this thesis, the keyboard and CRT display are used for interactive communication between user and controller. The microprocessor, RAM and few system routine facilities available on the processor card are used. The non-availability of non-vectored interrupt INTR at the edge connector restricts the number of interrupt lines for the user. Only three vectored interrupts, i.e. RST 5.5, RST 6.5 and RST 7.5 are available. The TRAP interrupt has been used inside the workstation. For servicing larger than three interrupt sources, software polling, which is slow, is the only solution. If INTR pin is also brought out to the edge connector, large number of interrupt sources can be efficiently serviced using programmable interrupt controller or hardware daisy chaining scheme.

## 5.2 HARDWARE DETAILS:

The hardware required for the controller of three phase bridge converter (Fig. 2.6) and the three phase pulsewidth modulator (Fig. 4.4) has been integrated on a single printed

circuit board. Provisions are kept for further expansion in the capabilities of the controllers. The sketch of chip layout and the IC's used is given in Fig. 5.2.

The common section for both the controllers is memory, both RAM and ROM, ports and workstation facilities. Both controllers have a interrupt based structure. Due to the availability of only three vectored interrupts the interrupt sources for both the controllers have to be multiplexed on to these interrupt lines. The details of the hardware description are given below.

#### 5.2.1 Memory:

The programs for the controllers are stored in EPROMS 2716. On board provision has been made for two of these EPROMS giving a total memory of 4K bytes. The chip selects for these EPROMS is obtained from decoder 74LS138 (Fig. 5.3). The address space selected for the EPROM is ~~0F000~~(Hex) to ~~0FFFF~~ (Hex). With changes in chip select logic and connecting  $A_{11}$  address line in place of  $V_{pp}$  (pin 21), EPROM 2732/2732A can be used giving a total ROM space of 8K bytes.

For the scratch pad memory required in the operation of the controllers, the user RAM provided inside the workstation has been used. The user RAM area inside the workstation is

from 4000H to 57FFH, of these less than 256 bytes are required by the controller.

#### 5.2.2 Timers:

The major function of both the controllers is the generation of timing waveforms with or without reference to external events. The timing events can be generated by software counting or hardware counting. In software counting, the microprocessor is all the while engaged in the counting process and cannot do any other function. For the controllers implemented in this thesis, the microprocessor has background functions to accomplish. Taking the example of pulsewidth modulator, even as pulsewidth count is being converted into delay, the microprocessor has to accomplish the task of computing pulsewidth count for next carrier interval and also to accomplish communication with user via keyboard and CRT display. Hence the relegation of counting to a hardware counter is a must.

The instant of conversion of a count into desired delay can be determined by software polling or by interrupt structure. For the same reason as given for using a hardware counter, the use of interrupt based structure is must.

On the printed circuit board developed (Fig. 5.2), provision for three programmable interval timers, (Intel 8253-5)

has been made. Timer 1 is used by the controller for the three phase bridge converter, timer 2 for extra provisions, timer 3 is used for three phase pulsewidth modulator. The timers are mapped in the I/O address space of the microprocessor. Timer 1 is addressed from 10 to 13 (Hex), timer 2 from 18 to 1B (Hex), timer 3 from 20 to 23 (Hex). The chip selects for the timers are derived from the I/O address decoder 74LS138, shown in Fig. 5.4. The functions of each timer are described below.

The Intel 8253-5, programmable interval timer contains three 16-bit programmable down counters. The operation of the counters is software selectable. For each counter a clock input, terminal count output and 'gate' pins are provided. Gate pin function depends on the mode of operation selected. The control byte and function of gate pin for any mode of operation can be obtained from Intel component data catalog.

#### Timer 1:

For the implementation of controller for three phase bridge converter, three counters are required as explained in Chapter 2. All the three counters have been implemented in one timer, timer 1 in Fig. 5.2.

The clock for all the counters of timer 1 is derived by dividing the microprocessor clock out by two using a flip-flop

(1/2 7476). Counter  $\phi$  is used to generate ' $\alpha$ ' delay. It is programmed in mode  $\phi$  or interrupt on terminal count, the gate of counter  $\phi$  (gate  $\phi$ ) is held high by connecting to supply. The terminal count out (out  $\phi$ ) is connected to interrupt multiplexing logic for RST 5.5 interrupt, explained later. Counter 1 is used to count zero crossing detection (ZCD) pulse, an input to the controller. Gate 1 has been provided at an edge connector, the circuit for zero crossing detection has to be externally configured. Out 1 is not connected (NC) to any interrupt. For counting the ZCD pulse at the gate, the counter can be programmed in many modes, mode 2 or rate generator mode is selected as it automatically reloads on positive transition. Counter 2 is programmed in mode 2 or rate generator mode. It is used to repeatedly count 60 degree interval. Gate 2 is connected to  $\bar{Q}_2$  at monostable (1/2 74LS221) which triggers on the positive edge of ZCD pulse to achieve synchronisation. The monoshot output pulse duration is 250 nsec, just sufficient to be recognised by 8253-5. Out 2 goes to interrupt multiplexing logic for RST 7.5.

#### Timer 2:

The counters in timer 2 are configured to provide expansion capabilities to the controllers implemented in this thesis or can be useful in implementing other controllers.



Counters  $\emptyset$  and 2 in this timer are clocked by microprocessor clock out by two, derived from same flip-flop as timer 1.

Counter  $\emptyset$  of timer 2 can be used for delay generation. Gate  $\emptyset$  is pulled high by connecting to supply. Out  $\emptyset$  is connected to RST 6.5 multiplexing logic. This can be used for generating delay, this is required for controller of cyclo-converters Ref.[4], may be required for modulation strategy of forced commutated converters.

Counter 1 is configured for counting events in a controlled time interval. The clock input pin for this counter has been provided on an edge connector. Hence events external to the controller can be fed to it. The gate of counter 1, gate 1 is derived from a multiplexing logic of AND-OR-INVERT (AOI) gates,  $PC_2$  control pin from PPI-1, the terminal count output pins of counter 2 (timer 2) and counter 1 (timer 3) see Fig. 5.5. The basic function achieved is that one of the counter outputs, OUT2 (timer 2) or OUT1 (timer 3) can be selectively connected to gate 1. Out 1 is not connected. The net result being that counter 1 can be made to count external events for a desired duration of time. This can be used as a closed loop counter to count say tachometer pulses for a speed feedback system.

Counter 2 is configured to generate delays. Gate 2 is pulled to supply, out 2 is connected to RST 6.5 multiplexing logic. When selected it will cause interrupt to denote the closed loop interval delay.

### Timer 3:

For the implementation of three phase pulsewidth modulator explained in Chapter 4, three counters clocked by selectable frequency is required. The three counters of timer 3 are configured for this. The clock for all the counters is derived from a monoshot (1/2 74LS221) output. The monoshot is triggered by inversion of terminal count output signal of a cascade of counters (74LS161A) clocked by high frequency source (24 MHz).

The gate pins for all the counters of timer 3 are connected to supply. Counter 0 output (OUT 0) is connected to RST 5.5 interrupt multiplexing logic, OUT 1 is connected to RST 6.5 multiplexing logic, OUT 2 is connected to RST 7.5 multiplexing logic.

During operation of the controller for three phase bridge converter all counters of timer 1 are in operation. During operation of three phase pulsewidth modulator all the counters of timer 3 are in operation. The time interval for

closed loop counting can be generated selectively by counter 2 timer 2 or counter 1 timer 3. When controller for three phase pulsewidth modulator is in operation, counter 2 of timer 2 should provide the closed loop timing interval. When controller for three phase bridge converter is in operation both can provide the closed loop interval time. But the maximum time interval generated by counter 2, timer 2 is 42.66 msec which can be small for control of converter dc motor drive system. The maximum time interval generated by counter 1, timer 3 is of the order of seconds infact 11 seconds which is definitely more than adequate.

### 5.2.3 Ports and Latches:

In the implementation of both the controllers in this thesis, the output trigger or gating pulses have been set at an output port. Quite a few control lines are required for proper operation of the controllers. All these are derived from the programmable peripheral interface (Intel 8255).

The programmable peripheral interface has three 8-bit ports. They are named as port A, port B and port C. They can be configured by software as input ports, output ports in different modes of operation. Port C is a special port which can be configured as half-input port and half as output port. For port A and/or port B in special mode of operation

(other than mode  $\emptyset$ ) few pins of port C have special significance. For details of control byte to select different modes, significance of port C pins for each mode refer to Intel component data catalog.

On board provisions have been made for two programmable peripheral interfaces PPI-1, PPI-2 (Fig. 5.2). These are mapped in the I/O address space of the microprocessor, PPI-1 is address from  $\emptyset8$  to  $\emptysetB$  (Hex), PPI-2 from  $\emptyset\emptyset$  to  $\emptyset3$  (Hex). The chip selects are derived from I/O address decoder 74LS138 see Fig. 5.4. The functions of the ports are described below.

PPI-1: Ports A and C of programmable peripheral interface-1 are configured as output ports. Port A is used to set output trigger pulses for thyristors. These lines go to input pins of line driver 74128. The port A drive capabilities is limited and it has to be buffered if the port is expected to drive signals over 5 feet of cable. The buffering selected is line driver 74128. These are ideally suited by impedance matching for sending signals over long cables. Their current sourcing capacity is also large, 20mA, this is useful for pulse isolation and amplification stage. The line driver 74128 is a NOR gate, hence outputs are inverted. Hence gating signals sent to port A have to be active low. The second input pins

of the NOR gates in line driver 74128 are obtained from a SPDT switch connected between supply and ground. The state of the switch is indicated by an led (green). When the switch connects the inputs of the NOR gates to ground, outputs of the NOR gates reflect complemented input gating signals. When all inputs are connected to '1' (supply) all the outputs are permanently low thus disabling gating pulses. Hence the switch can be called output enable for gating pulses. Infact if the controller has added features of fault monitoring, this signal with appropriate gating could be connected to the second input of NOR gates of 74128, resulting in automatic disabling of output pulses upon the occurrence of fault. Then the micro-processor in its fault routine (by polling) can diagnose the fault.

Port B of PPI-1 is configured as input port. It is connected to eight output lines of a 8-bit analog to digital converter. Hence port B is read by the controller to read the digital equivalent of analog control input.

Port C of PPI-1 is configured as an output port. It generates all the control signals for proper multiplexing of interrupts, multiplexing of input to gate 1 timer 2 (sec. 5.2.2), multiplexing analog input to ADC etc. The details are given below.

$PC_0$  is used for interrupt multiplexing for ZCD monoshot output and counter 2, timer 1 for RST 7.5 interrupt. The interrupt multiplexing logic is shown in Fig. 5.8.

$PC_1$  is used to select bank of interrupt source to be connected to host interrupt lines.

$PC_2$  is used as control signal for multiplexing OUT2 timer 2 or OUT 1 timer 3 to gate 1 as explained in Sec. 5.2.2.

$PC_3$  and  $PC_4$  are used to set the flip-flops of 7476 high. The clear input to these flip-flops are brought out to the edge connector of the printed circuit developed. The outputs of the flip-flops  $Q_1$ ,  $Q_2$  are connected to another port from where they can be read to determine state of  $Q_1, Q_2$ . Hence, by software polling, positive transitions in external events can be sensed. They can be used in the controller for single phase non-circulating current type cycloconverter to sense the zero crossings of output current (Ref. [4]). The entire logic for this has to be external. Such a scheme has to be adopted due to the acute shortage of interrupt lines available to the user from the host system.

$PC_5$  to  $PC_7$  are connected to analog multiplexer (HEF 8051) to select one of the eight channels to be input to the analog to digital converter.

PPI 2: The ports of PPI-2 are actually meant for expansion capabilities. If all the expansion facilities on the printed circuit are eliminated, including flip-flops for external events, multiple inputs to ADC, the few lines of port C which are required for operation of controller for three phase bridge converter can be shifted onto PPI-1 port C lines and then PPI-2 is not required.

Port A is unused but the lines are connected to sockets configured for line drivers 74128. A controller for single phase cycloconverter will require this to set output trigger pulses for additional inverse parallelly connected converter. It will also be required for controller of converter fed four quadrant dc drive.

Port B is unused and provided at the edge connector of the printed circuit card. It can be used both as input or output port.

Port C only half has been used, the other half has been brought to the edge connector for expansion purposes explained below.

$PC_0$  is connected to OUT  $\emptyset$  timer 2. It is used as software polling means to identify the source of RST 6.5 interrupt.

$PC_1$  is connected to end of conversion (EOC) line of the ADC. By software polling the completion of analog to digital conversion is sensed before reading port B, PPI-1 for digital count.

$PC_2$ ,  $PC_3$  are used to sense the state of  $Q_1$ ,  $Q_2$  of 7476-2 used to detect the positive transitions of external events by software polling.

Port C lines  $PC_4$  to  $PC_7$  in conjunction with port B can be used with multiplexers for getting various fault detection signals by software setting and reading operation. (Ref.[6]). Thus a large number of sensing points can be selectively read by the microprocessor.

Two additional 8282 output latches are presented on the card. The use and functioning of these are explained in Section 5.2.6.

#### 5.2.4 Analog to Digital Converter with Analog Multiplexer:

Analog to digital converter can be used as control input device, to read in system feedback variables. The speed of conversion is important as it affects the performance of the controllers. In the developed software it is used only by by controller for three phase bridge converter.

The ADC selected is a National semiconductor, successive approximation analog to digital converter, ADC 0800. It requires



bipolar supply. The conversion time is 50  $\mu$ sec for 300 KHz external conversion clock. The ADC has been configured to accept bipolar input voltage in the range +5v to -5v. The power supply required is +5v and -12v, a -5v required by the converter has been generated on the card itself. The bipolar input nature is advantageous for analog to digital conversion of feedback variables.

The outputs of the ADC are connected to port B PPI-1. The conversion clock is derived by dividing the microprocessor clock out by four. It is derived from  $Q_2$ , 7476. Flip-flop  $Q_2$  of 7476 is clocked by output of flip-flop  $Q_1$  which is clocked by microprocessor clock out. The start conversion pulse is obtained by inverting the signal obtained at one of the memory decoder 74LS138 pins. Hence a instruction STA  $\emptyset D \emptyset \emptyset \emptyset$  (Hex) or LDA  $\emptyset D \emptyset \emptyset \emptyset$  (Hex) gives the required start pulse to the ADC.

The analog input to the ADC is obtained from the output of a analog multiplexer of 8 input channels. The supply to the analog multiplexer is +5v, -5v. The channel select signals set by  $PC_5 - PC_7$  of PPI-1. All the eight input lines of the analog multiplexer have been brought to the edge connector of the printed circuit board.

#### 5.2.5 Variable Frequency Clock Generation:

A high resolution, controllable frequency clock is required for the operation of three phase pulsewidth modulator

as explained in Section 4.3.1. A digital implementation for it involves a count down (or up) of a variable, selectable count by counters clocked by higher frequency clock source.

The high frequency clock source is obtained from a crystal oscillator of 24 MHz with the circuit of two inverting gates and a trimpot capacitor (see Fig. 5.6). The inverting gates of 74S04 have been used for this purpose, trimpot capacitor value selected is 20 pF.

The counter selected is 4-bit UP counter, 74LS161A. An entirely synchronous counter except for the asynchronous clear which is not used. A cascade of four 74LS161A is present on the printed circuit board. The carry-look ahead scheme of cascading is used (Fig. 5.7). The selectable counting feature is incorporated by using the load facility.

The carry out pulse of each stage of counter is of a input clock period duration i.e. 41.7 nsec. It enables the counting for the next stage. The carry out pulse of the most significant counter denotes that the carry out of all stages has occurred. This is connected to 'load' pin of all the stages after inversion. When load pin goes low, the count set at the input of each counter gets set up at the counter flip-flops synchronously. Thus by selectively setting input both the counters and implementing the cascade stage with

ripple carry out of final stage connected to load input pins of all stages, variable counting can be achieved. Thus the ripple carryout frequency becomes variable. This is the desired variable frequency clock. This is stretched by monoshot 1/2 74LS221 before being fed to clock inputs of all counters in timer 3.

The variable count for the counters is set at the input of the counters by two eight bit 8282 latches. The inputs to the latches is data bus signals, the strobe signal is derived from a chip-select,  $\overline{WR}$  with a NOR gate. Hence  $STB = \overline{CS} + \overline{WR}$ . The chip selects are derived from I/O address decoder. The lower byte 8282 chip select is 28H, most significant byte 8282 chip select is 4 30H. Thus, the count to be latched at lower byte 8282 can be sent by a OUT 28H instruction.

The counters 74LS161A are selected for its synchronous operation which results in final ripple carry out at one input clock duration. This in conjunction with the 24 MHz clock results in carry out pulse duration of 41.7 nsec, just above 40 nsec minimum trigger pulse duration required for monostables. The monostable output is selected as 250 nsec, the minimum clock duration for a clock to Intel 8253 programmable interval timer.

The selected counter i.e. 74LS161A and the counter 74LS163A are pin compatible. Since clear facility is not used in the design, either of them can be used. The 8282 are non-inverting latches, 8283 latches are inverting latches, both are pin compatible. In Sec. 4.2.2 it has been explained that count to be latched at the 8282 latches has to be complemented before being sent out, by using 8283 latches, the complimenting operation can be avoided.

#### 5.2.6 Interrupt Multiplexing:

Due to the interrupt structure adopted with large number of interrupt sources and limited number of interrupt available on the host system especially due to nonavailability of INTR interrupt, multiplexing of interrupt becomes a must.

Interrupt sources of the controller for three phase bridge converter are independent of the interrupt sources for three phase pulsewidth modulator. Hence two banks of interrupts for the two circuits are multiplexed onto the vectored interrupt lines using digital quad 2-bit multiplexer 74157. The outputs of 74157 are inverted before being connected to host signals. The bank select control line is derived from  $PC_1$ , PPI-1. Where multiplexing on a single interrupt line for a circuit becomes must, the multiplexing is done using AND-OR-INVERT gates (AOI) 7451. This results in reduced chip count.

The control signals are derived from port C, PPI-1. The circuit diagram is shown in Fig. 5.8.

### 5.3 DESCRIPTION OF AUXILLARY HARDWARE DEVELOPED:

#### 5.3.1 Pulse Isolation and Amplification Stage:

The gating signals have to be amplified to be compatible with gate requirements and isolation is required between the low power control circuit and high power power electronic circuit. In the pulse isolation and amplification stage double isolation is provided. The first isolation is provided by optocoupler 6N136, second by pulse transformer. The input to optocoupler should come from a gating pulse capable of sourcing 5 mA of current. The optocoupler has been configured as emitter follower. The optically isolated pulse goes to the reset input of NE555, the output of NE555 represents the gated waveform of input duration. There is no delay angle ( $\alpha$ ) error due to gating process. In case of AND operation of trigger pulse and astable clock of around 10 KHz to obtain gated waveform, an error of 50 microsecond is possible. This is due to the unsynchronisation of trigger pulse output and astable gating waveform. An inverting transistor amplification stage is used ahead Fig. 5.9. A pulse transformer is connected in the collector circuit to give second stage of isolation. The power supply to each stage can be seperated. The PCB layout is given in Appendix A.

Pulse transformer is a bipolar device, hence backward reflection of power circuit transients is possible. But optocoupler is a unipolar device, hence absolute isolation is guaranteed. If only optocoupler isolation is provided, then as the gate, cathode will have to be connected across output and ground of optocoupler; ideally each optocoupler amplification stage requires an isolated power supply. It is a must atleast for all the power switches in the top group of thyristors.

#### 5.3.2 Optical Isolation Zero Crossing Detector:

The zero crossing detector is required to get zero crossing instances in the line to line waveform. This helps in synchronisation and operation of the controller for three phase bridge converter.

The zero crossing detector has been designed using a optocoupler 6N136. The input stage of a optocoupler is a photodiode which is optically coupled to a output phototransistor. When photodiode conducts the phototransistor turns ON. A 15K , 20W wire wound resistance is connected in series with the photodiode to restrict the current through the photodiode. A diode is connected inverse parallely to the photodiode, across this entire circuit the line to line voltage is applied as shown in Fig. 5.10. When the photodiode conducts

the output becomes low. When voltage polarity reverses, the inverse parallelly connected diode conducts, resulting in a inverse voltage of 0.7V, less than 5V max PIV acceptable by a photodiode, thus protecting it. The series current limiting resistance is selected to restrict peak current through the photodiode to below the maximum acceptable value of 40 mA.

### 5.3.3 Analog Signal Optical Isolation:

The isolation of feedback signals is a must. With advances in optoelectronics, solid state, low volume, low cost optical couplers for isolation of analog signals are available. The circuit for a analog signal optical isolation stage is given in Fig. 5.11.

The optical couplers 4N45 have linear input-output characteristics for a certain range of photodiode forward current. The current through the photodiodes for optocouplers 1 and 2 is controlled by a closed loop feedback system. The supply voltages  $\pm V_{cc1}$  to transistors  $T_1, T_2$  and optocoupler 2 is independent and also isolated from the supply  $V_{cc2}$  to optocoupler 1. By variations in input voltage, the current through the photodiodes varies linearly resulting in linear

variation at the output of optocoupler 1. Thus isolation of even dc signals is easily possible.

A common PCB has been made for ZCD and analog signal optical isolation stages. The layout is given in Appendix A.



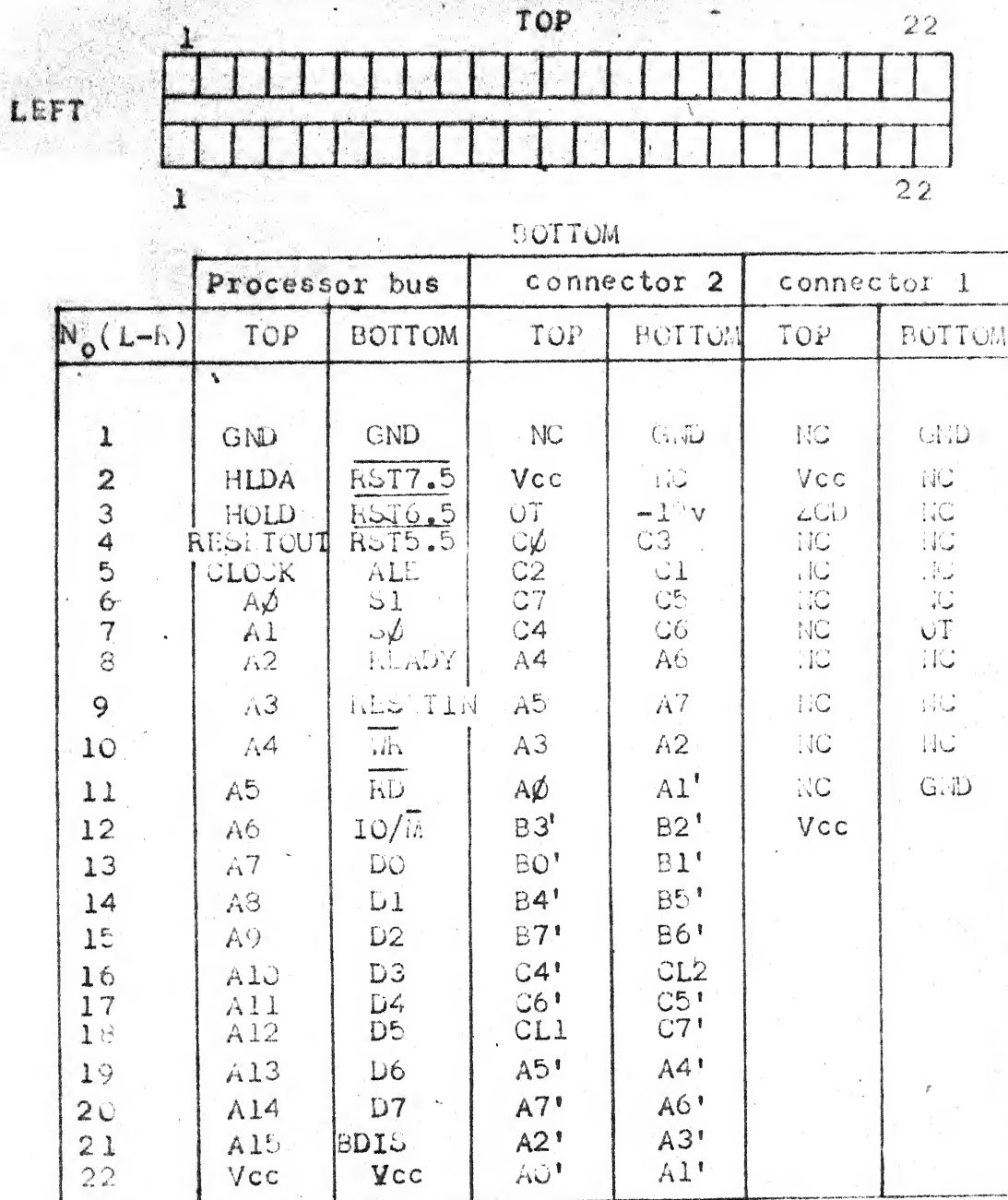


Fig. 5.1: Connector Signal Diagram.

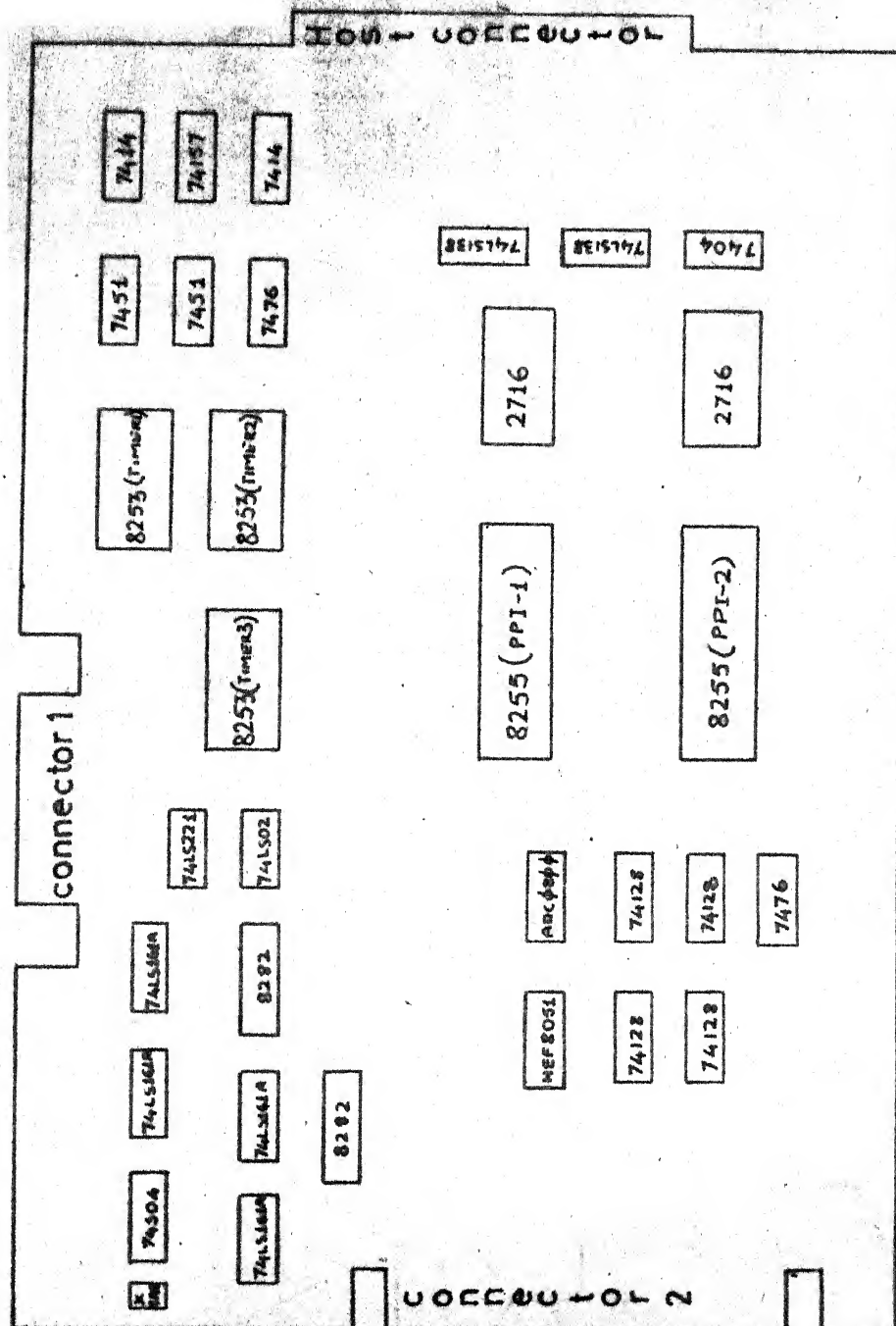


Fig 5.2 Sketch of ICS on the PCB

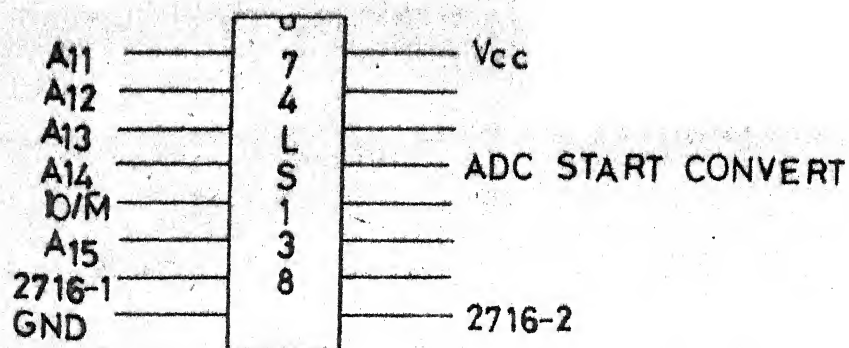


Fig 5.3 Memory Address Decoder

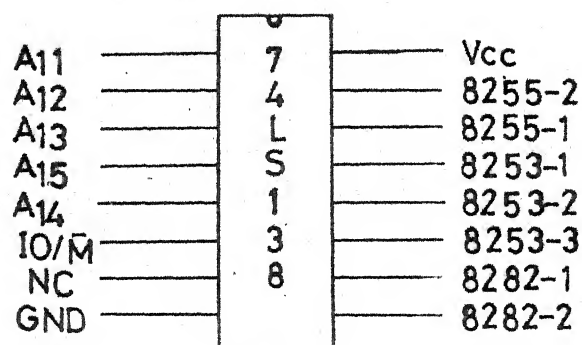


Fig 5.4 I/O Address Decoder

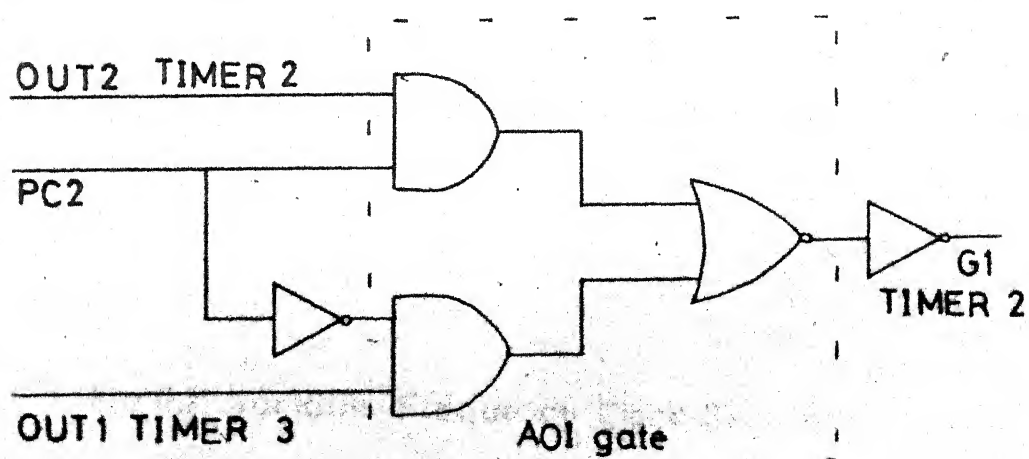


Fig 5.5 Multiplexing for G1 Timer 2

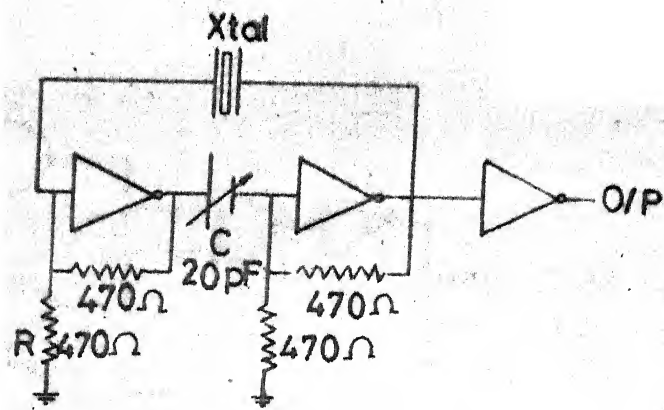


Fig 5.6 Crystal Oscillator Circuit, Gates are 74S04

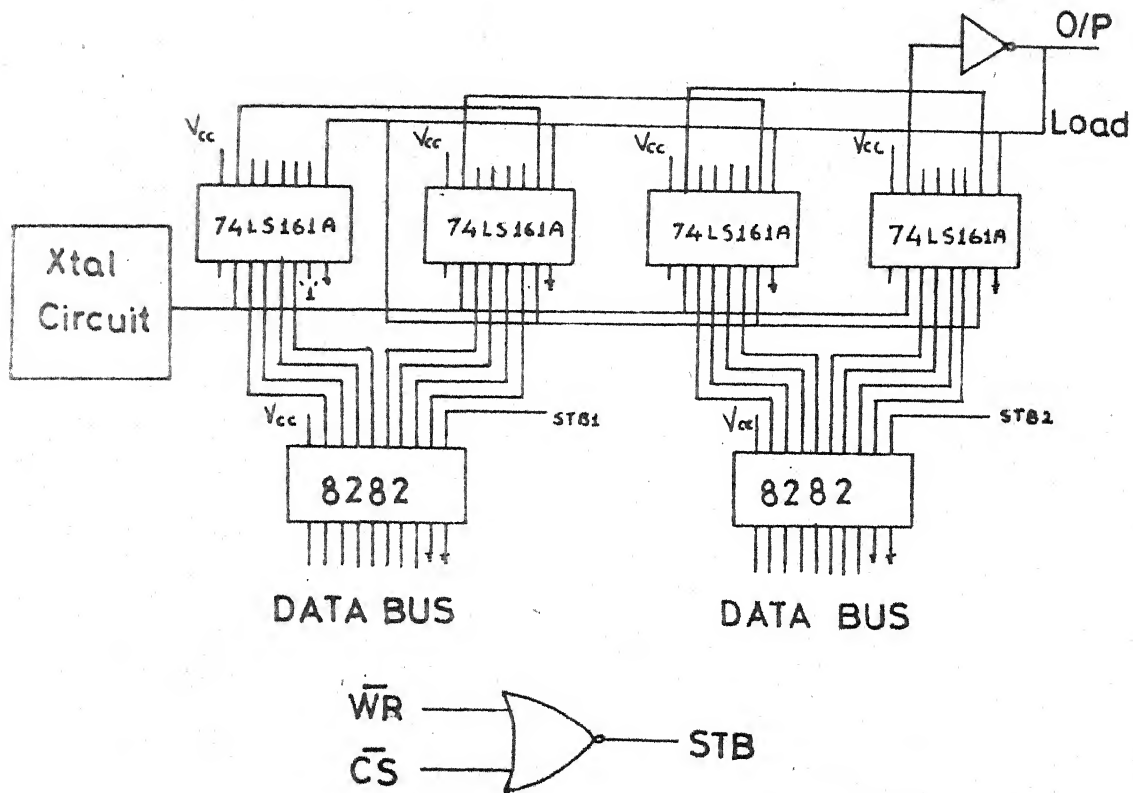


Fig 5.7 Variable Frequency Clock Generator

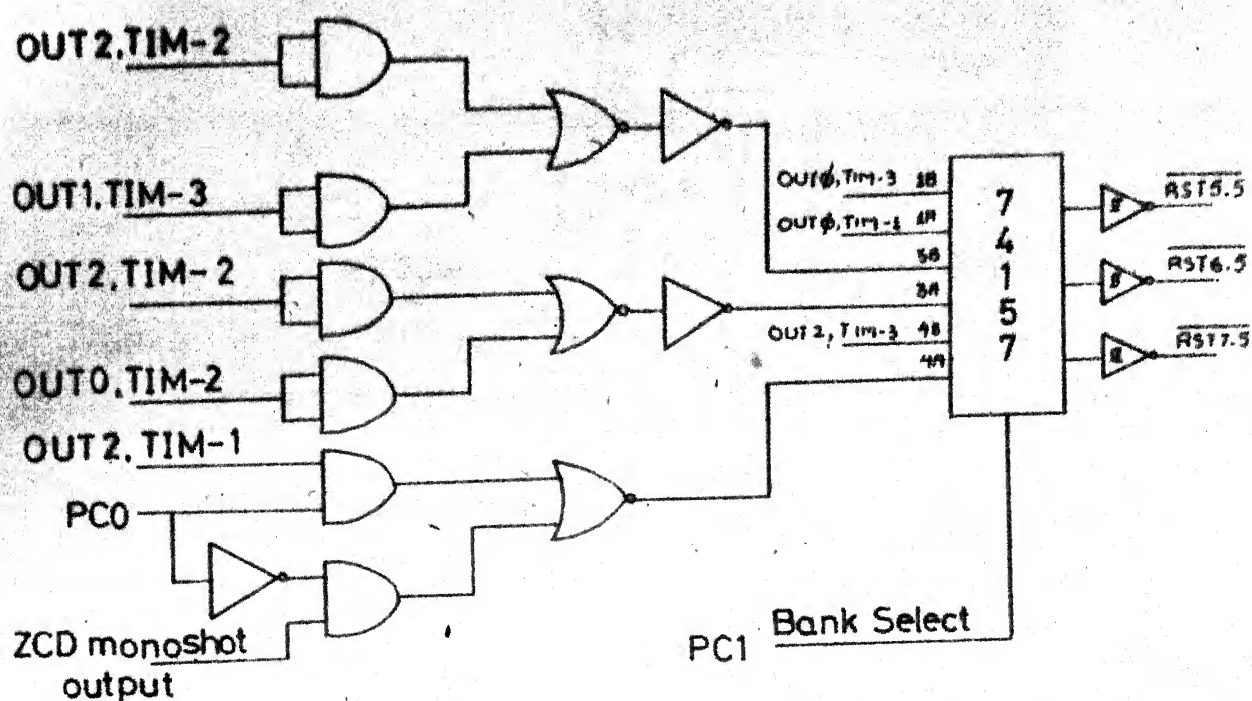


Fig 5.8 Interrupt Multiplexing Logic

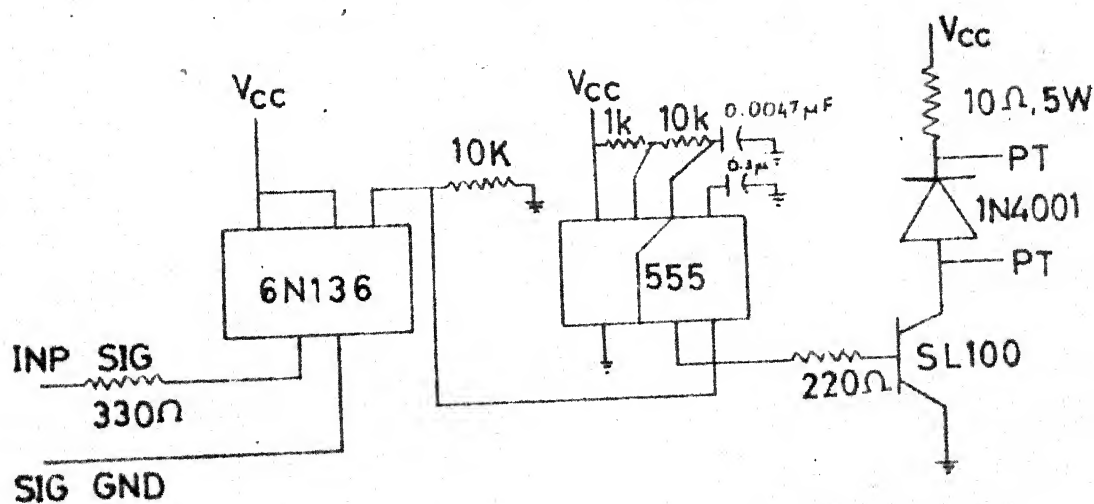


Fig 5.9 Pulse Isolation and Amplification

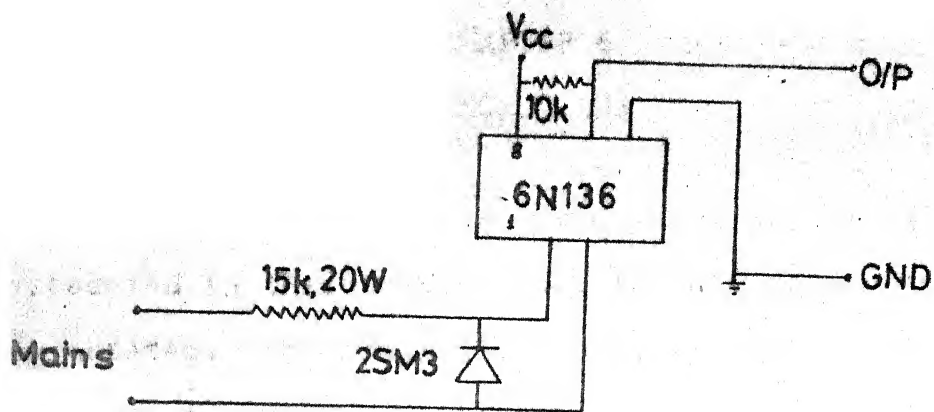


Fig 5.10 Optocoupler based ZCD

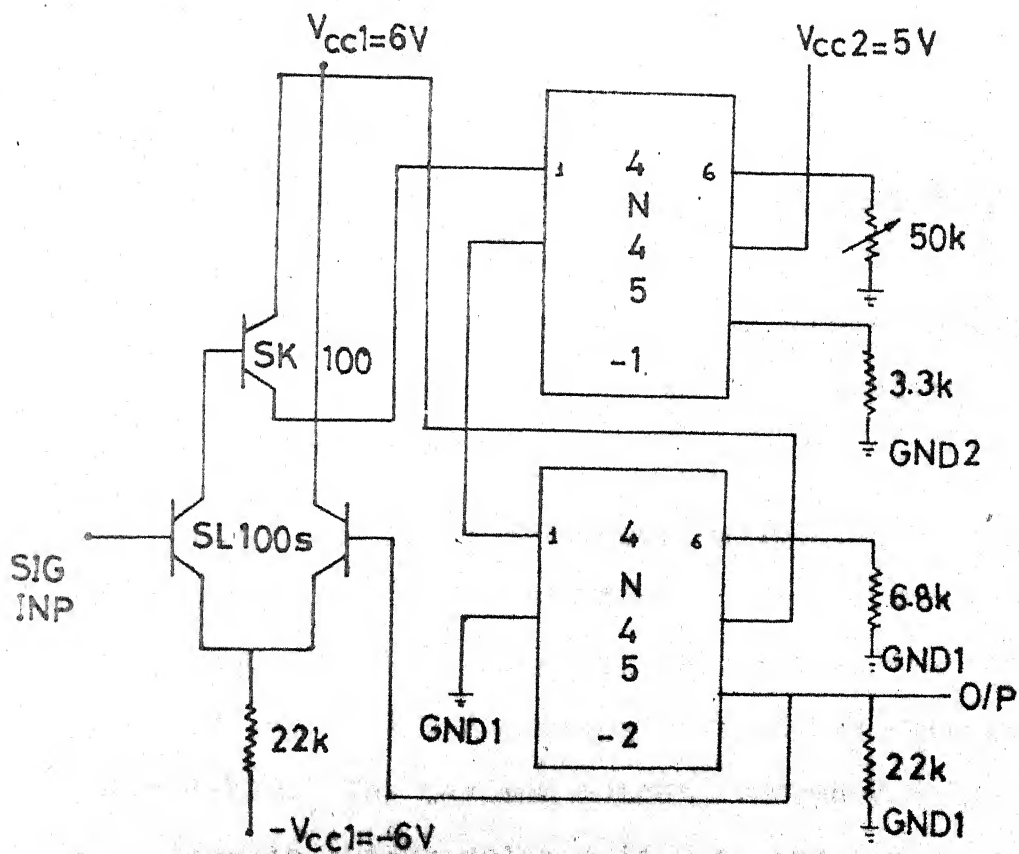


Fig 5.11 Analog Signal Optical Isolation

## CHAPTER 6

### CONCLUSIONS

The controller for three phase bridge converter presented in this thesis has a few advantages these are enumerated. The controller requires only a single phase sensing and hence equidistant triggering is ensured, Ref. [5] uses three phase sensing. Synchronisation is achieved by use of programmable timer 8253 alone, Ref. [4,6] use hardware PLL logic to achieve synchronisation. The status of supply voltages is determined entirely by software, in Ref.[5,6] the status of supply voltages has to be read via a port. The use of modified delay and only one delay counter results in fast transient response of  $1/6$ th cycle, this is not possible for schemes in Ref.[1,2,3,4]. The controller for three phase bridge converter implemented in this thesis exploits the software manipulative capabilities of the microprocessor and results in reduced hardware.

The three phase pulsewidth modulator implemented in this thesis uses a multimode strategy. Independent control of voltage and frequency is possible. But to demonstrate an open loop system a V/F characteristics has been stored in a lookup table. The maximum output frequency is 100 Hz. The resolution in output voltage is 0.4% and output frequency is 0.2 Hz.

In the low frequency range of 0 to 20 Hz, a carrier by carrier computation based sinusoidal PWM is implemented. Above this range, optimal PWM is implemented. The optimal PWM transits through pulse dropping stage to square wave mode at maximum output voltage which occurs at rated output frequency of 50 Hz.

In the computation scheme adopted in SPWM, the pulse-width counts are a function only of the output voltage. Hence there is a decoupling of voltage and frequency. This is essential for optimal PWM, harmonic elimination scheme. This decoupling is absent for the computation scheme presented in Ref.[9]. Hence with scheme given in Ref.[9] implementing a multimode strategy is not possible.

Within SPWM range, asynchronous SPWM is implemented for 0 to 5 Hz, synchronous SPWM with ratio 36 for 5 to 10 Hz and synchronous SPWM with ratio 18 for 10 to 20 Hz. A Hysteresis band of 1 Hz is provided between adjacent modes. Smooth and efficient transition between adjacent modes is ensured.

#### Suggestions for Further Improvements:

In the controller for three phase bridge converters if a constant frequency supply with minor variations is assured, the software computations can be reduced by just computing the errors, for 60 degree interval determination.



Instead of dividing half period time, the change in half period time can be used to determine the corresponding change required in 60 degree interval. The arc cosine table can store the delay in number of clock states and correction for minor variations can also be done to the delay count before loading into the delay counter.

Since synchronisation with ac supply is available and provision kept for further development, the software required for control of three phase modified bridge and single phase cycloconverter can be developed. With only software changes controller for single phase bridge converter can be developed.

In the three phase pulsewidth modulator, the execution time delay problem can be overcome by using edge sensitive interrupts and hardwired logic for generating output pulses. Since three edge sensitive interrupts are not available, RST 5.5 and RST 6.5 can be made to behave as edge sensitive by having flip-flops connected to them. The interrupting source should set the flip-flops and the microprocessor in the respective ISS should reset it using control lines of a port. The hardware logic required for generating output pulses consists of a port and synchronising flip-flops clocked by terminal count outputs of the pulsewidth counters.

This, if possible should be rigged on the card or jumpers be used to configure it externally or develop a new printed circuit board with these provisions.

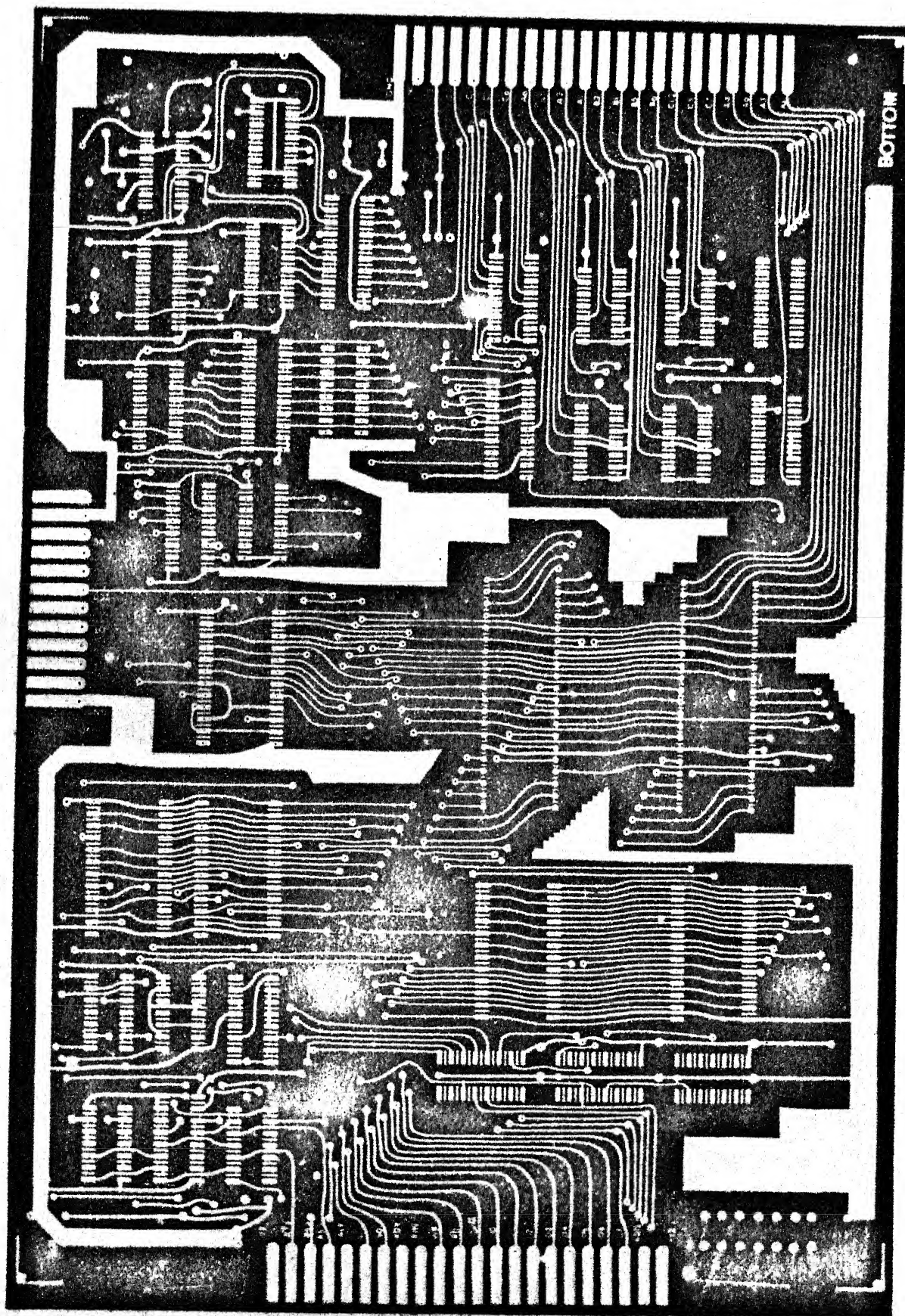
If very large ratio of carrier frequency to modulating frequency is required it would be desirable to implement a similar scheme with a faster microprocessor in which multiplication is available as a software instruction.

## REFERENCES

1. S.B. Dewan, W.G. Dunford, 'A Microprocessor-based Controller for a Three-phase Controlled Rectifier Bridge', IEEE Trans. IA, IA-19, Jan./Feb. 1983, pp. 113.
2. S.K. Tso, P.T. Ho, 'Dedicated-microprocessor scheme for thyristor phase control of multiphase converters', IEE Proc., vol. 128, Pt. B, No. 2, March, 1981, pp. 101.
3. G. Oliver, V.R. Stefanovic, G. April, 'Microprocessor Controller for a Thyristor converter with an Improved Power Factor', IEEE Trans. IECI, vol. IECI-28, No. 3, Aug. 1981, pp. 188.
4. S.K. Tso, E.D. Spooner, J. Cosgrove, 'Efficient micro-processor based cycloconverter circuit', IEE Proc. vol. 127, Pt. B, No. 3, May, 1980, pp. 190.
5. P.C. Tang, S.S. Lu, Y.C. Wu, 'Microprocessor-based Design of a Firing Circuit for Three Phase Full-wave Thyristor Dual Converter', IEEE Trans. IE, vol. IE-29, No. 1, Feb. 1982, pp. 67.
6. Y.V.V.S. Murthy, 'Microprocessor based Control and Fault Diagnosis of a Three Phase Bridge Converter', M.Tech. thesis submitted to EE Dept., IIT/K, Feb. 1983.
7. K.P. Gokhale, G.N. Revankar, 'Microprocessor-controlled separately excited DC-motor drive system', IEE Proc, vol. 129, Pt. B, No. 6, Nov. 1982, pp. 344.
8. S.K. Tso, F.W. Pu, 'Software realisation of synchronisation and firing control of thyristor converters', IEE Proc., vol. 131, Pt. B, No. 4, July 1984, pp. 141.
9. S.R. Bowes, M.J. Mount, 'Microprocessor Control of PWM Inverters', IEE Proc., vol. 128, Pt. B, No. 6, Nov. 1981, pp. 293.
10. D.A. Grant, 'Techniques for pulse dropping in pulse-width modulated inverters', IEE Proc, vol. 128, Pt. B, No. 1, Jan. 1981, pp. 67.
11. D.A. Grant, R. Seidner, 'Technique for pulse elimination in pulsedwidth modulated inverters with no waveform discontinuity', IEE Proc., vol. 129, Pt. B, No. 4, July, 1982, pp. 205.

12. D.A. Grant, R. Seidner, 'Ratio changing in pulsewidth - modulated inverters', IEE Proc., vol. 128, Pt. B, No. 5, Sept. 1981, pp. 243.
13. V.P. Ramamurthi, B. Ramaswami, 'A Novel Three-phase Reference sine-wave Generator for PWM Inverters', IEEE Trans. IE, vol. IE-29, No. 3, Aug. 1982, pp. 235.
14. B.K. Bose, H.A. Sutherland, 'A High-performance Pulsewidth Modulator for an Inverter-fed Drive System using a Microcomputer', IEEE Trans. IA, vol. IA-19, No. 2, Mar/April 1983, pp. 235.
15. G.S. Buja, J.B. Indri, 'Optimal Pulsewidth Modulation for feeding AC Motors', IEEE Trans. IAS, vol. IA-13, No. 1, Jan./Feb. 1977, pp. 38.
16. G.S. Buja, P. Fiorinni, 'Microcomputer Control of PWM Inverters', IEEE Trans. IE, vol. IE-29, No. 3, Aug. 1982, pp. 212.
17. K.S. Rajashekara, J. Vithayathil, 'Microprocessor based sinusoidal PWM Inverter by DMA transfer', IEEE Trans. IE, vol. IE-29, No. 1, Feb. 1982, pp. 46.
18. T.L. Grant, T.H. Barton, 'Control Strategies for PWM Drive', IEEE Trans. IA, vol. IA-16, No. 2, Mar/Apr. 1980, pp. 211.
19. H.S. Patel, R.G. Hoft, 'Generalised techniques of harmonic elimination and voltage control in thyristor inverters: Part I - Harmonic elimination', IEEE Trans. IA, vol. IA-9, May/June 1973, pp. 310.
20. H.S. Patel, R.G. Hoft, 'Generalised techniques of harmonic elimination and voltage control in thyristor inverters: Part II - voltage control techniques', IEEE Trans. IA, vol. IA-10, Sept/Oct. 1974, pp. 666.
21. V.V. Deshpande, 'Microprocessor based PWM Modulator for Three Phase Induction Motor Drive', M.Tech. thesis submitted to EE Dept, IIT/K, Sept. 1984.
22. B.K. Bose, 'Adjustable speed AC Motor Drive Systems', IEEE Press Book, 1981.
23. W. Bosshart, 'Printed Circuit Boards: Design and Technology', CEDT Series, TMH 1983.

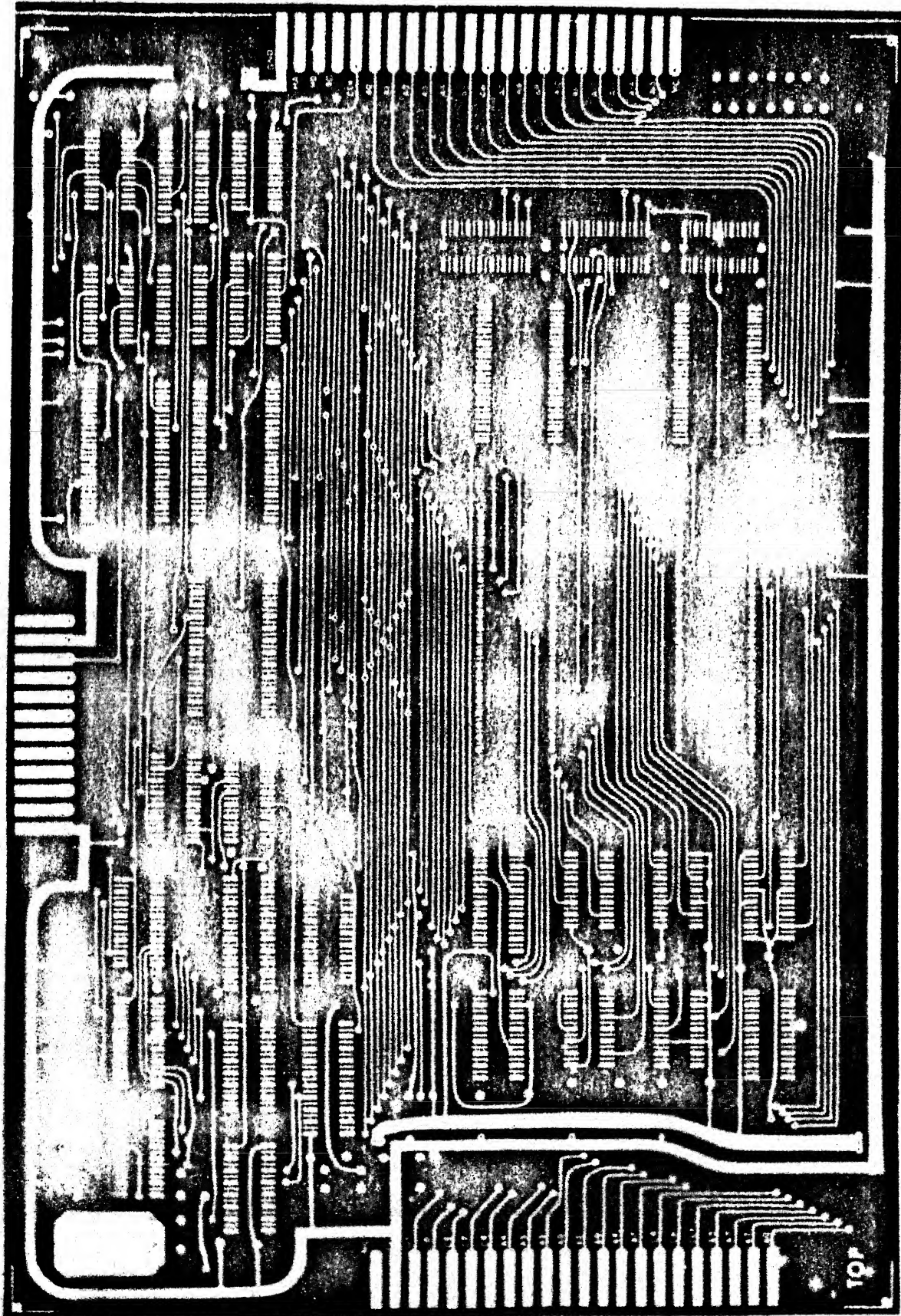
```
*****  
*APPENDIX-A : PRINTED CIRCUIT LAYOUTS *  
*FOR THE CONTROLLER AND ISOLATION PCB'S*  
*DEVELOPED *  
* *  
*DEVELOPED BY: MUKUND L. GHONASGI *  
*GUIDE: DR. A. JOSHI *  
*YEAR: 1983-1984 *  
*****
```



A-1

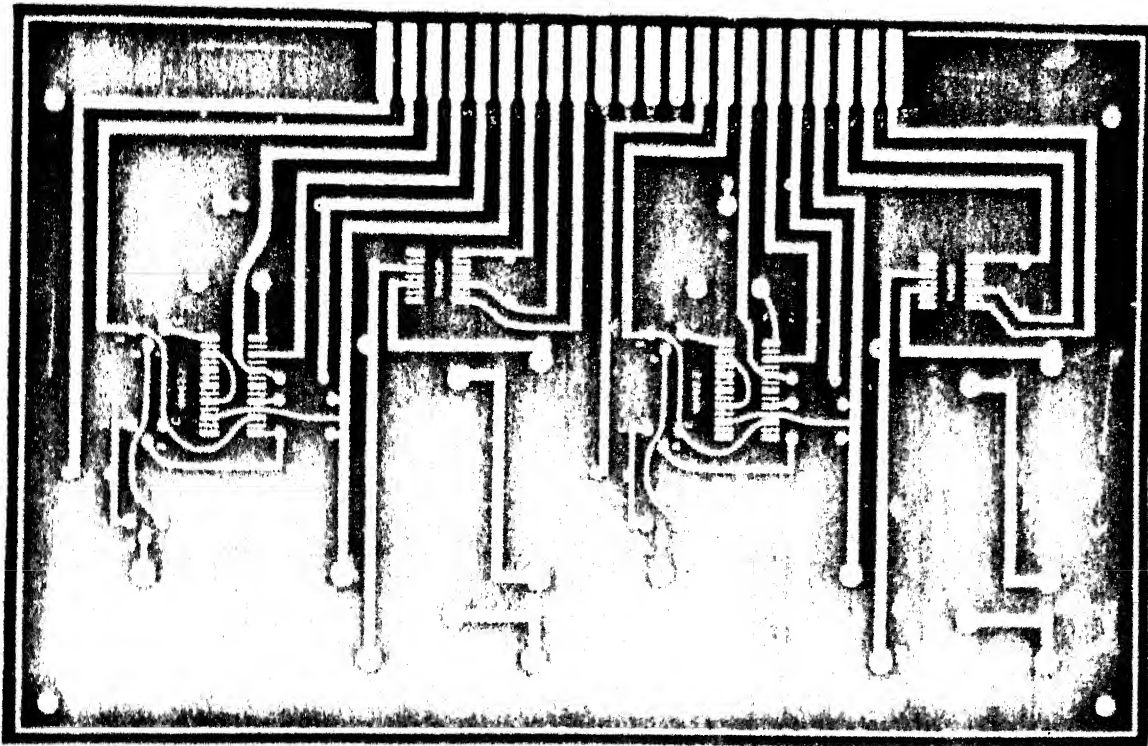
Layout of Bottom Side of Controller PCB



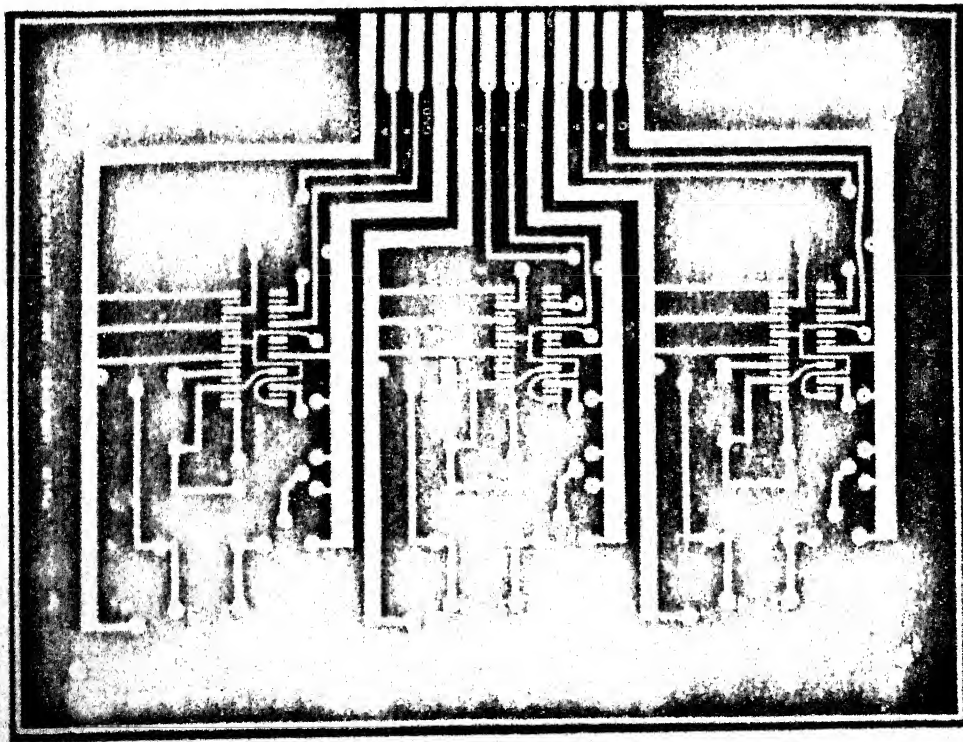


A-2

Layout of Top Side of Controller PCB

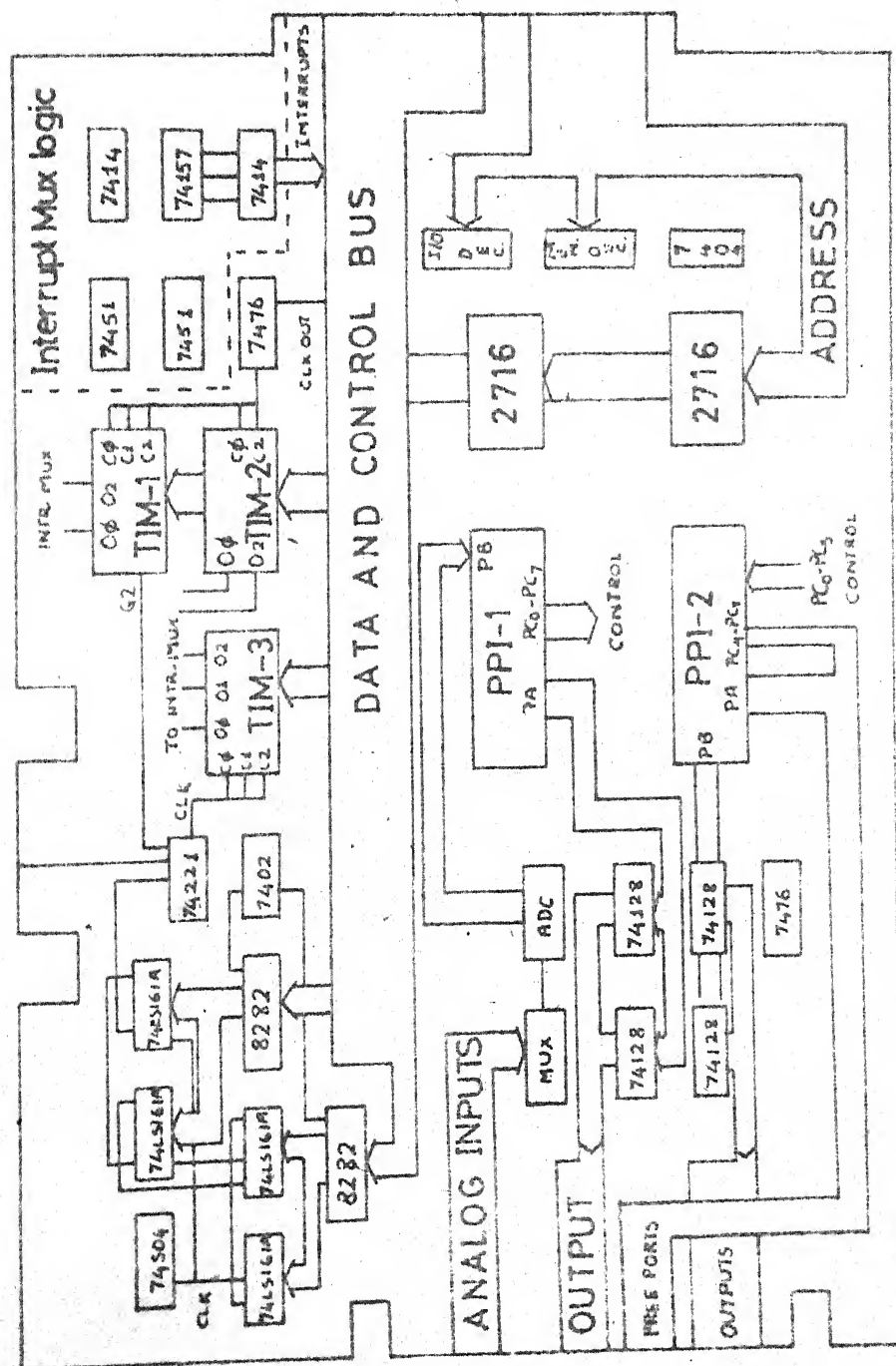


PCB Layout of ZCD Analog signal Isolation



PCB Layout: Pulse Isolation and Amplification





### Schematic Hardware Connection Diagram: